



UNIVERSITY^{OF}TARTU

FACULTY OF MATHEMATICS AND COMPUTER
SCIENCE

Institute of Computer Science
Software Engineering Curriculum

Zurab Bzhalava

**Exploring the animal GPS system: a machine learning
approach to study the hippocampal function**

Master's Thesis (30 ECTS)

Supervisor: Raul Vicente Zafra

Tartu 2015

Exploring the animal GPS system: a machine learning approach to study the hippocampal function

Abstract

The 2014 Nobel prize in Physiology was awarded to Dr. John M. O’Keefe, Dr. May-Britt Moser and Dr. Edvard I. Moser for discovering particular cells in the brain that provide the sense of place and navigation. These discoveries suggest that the brain creates internal map-like representation of the environment which helps us recognize familiar places and navigate well. In this thesis, we used a computational approach to study the animal "GPS" system. In particular, we set to compare how well different machine learning algorithms are able to predict a rat’s position just based on its hippocampal neural activity. Methods compared included Random Forest, Support Vector Machines, k-Nearest Neighbors, and several sparse linear regression algorithms. Data was obtained multi-neuron electrophysiological data recorded from the Buzsaki lab in New York, and we focus on the activity of rat hippocampus, the brain region where most the place cells have been identified.

In a first step, we divided the experimental arena into 4 blocks and tried to classify in which one of those blocks the rat was at a given time. In this case, we found that Random Forest gave the best accuracy which was 57.8%, well beyond the chance level. However, in some particular regions of the arena, Support Vector Machine was sometimes better than Random Forest. For the next step, we made the classification problem even harder by dividing the arena into 16 blocks. Random Forest and SVM produced highly significant results with 38% and 37% accuracy respectively (random classifier accuracy would be approximately 11%). We also used K-Nearest Neighbors for both classification problems but its accuracy was less in both cases than the above mentioned algorithms. Since the rat position is a continuous variable we also considered the continuous prediction problem. Most regression algorithms we analyzed (Ridge Regression, LASSO, Elastic Net) provided results near chance level while Random Forest outperformed the algorithms and gave the best results in this case.

Furthermore, we analysed data recorded from an experiment where rats were trained to choose left or right direction in a 8-shaped maze while they were running in a wheel. In this case we perform a dimensionality reduction of the neuronal data to visualize its dynamics during the decision time. We also identified and provided plots of episodic cells (neurons who are more active at particular times in the task) which might contribute to the sense of time and create episodic memory. Also, we visualized neuronal trajectories while animal makes decisions in order to predict its future decision.

In conclusion, from the algorithms we analysed Random Forest gave the best accuracy while predicting a rat’s location. This might also indicate that the information about rat location is contained in non-linear patterns of neuronal activity, which linear regression methods were unable to extract. In future research we plan to decode a

rat position using a method more similar to the brain own mechanisms such as neural networks, which as Random Forest can detect non-linear patterns. More generally, the pipelines developed during this thesis to handle the complex pre-processing, feature extraction, and visualization of the dataset will set the basis for future studies on hippocampal dynamics by the group of computational neuroscience in the University of Tartu.

Key Words: hippocampus, neural data analysis, machine learning, place cells, decoding

Looma GPS“ süsteemi uurimine: Masinõppimise kaudu hipokampuse funktsiooni tundma õppimine

Kokkuvõte

2014. aasta Nobeli preemia füsioloogias said Dr. John M. O’Keefe, Dr. May-Britt Moser ja Dr. Edvard I. Moser teatud kindlate rakkude avastamise eest ajus, mis vastutavad ruumi- ja suunataju eest. Need avastused võimaldavad arvata, et aju loob sisemise kaardi ümbritsevast keskkonnast. See aitab meil ära tunda tuttavaid kohti ja ruumis hästi orienteeruda. Antud magistritöös kasutasime me roti “GPS” süsteemi tundma õppimiseks arvutuslikku lähenemist. Konkreetsemalt võrdlesime, kui hästi suudavad erinevad masinõppe algoritmid ette ennustada roti asukohta, saades sisendiks ainult tema hipokampuses toimuva neuronaalse aktiivsuse. Võrreldud meetodite seas olid juhums (random forest), tugivektorklassifitseerijad (support vector machine, SVM), lähima naabri meetod (nearest neighbor) ja mõningad hajusa lineaarse regressiooni algoritmid. Neuronitest mõõdetud elektrofüsioloogilised andmed olid pärit Buxsaki laborist New Yorgis. Keskendusime roti hipokampuse neuraalsele aktiivsusele - aju osale, kus on senistes uurimustöodes enamik koharakke tuvastatud.

Esimese sammuna jagasime ala, kus rott eksperimendi ajal viibis, neljaks väiksemaks tsooniks. Seejärel üritasime ennustada, missuguses alas katsealune loom mingil suvalisel ajahetkel asus. Leidsime, et juhums andis parima ennustustäpsuse, milleks oli 57.8% ja mis on oluliselt suurem juhusliku valiku tõenäosusest. Sellegipoolest oli mõnedes katseala regioonides tugivektorklassifitseerija mõnikord parem kui juhums. Järgmise sammuna tegime asukoha identifitseerimise veelgi raskemaks ja jagasime eksperimentaalala 16 väiksemaks tsooniks. Juhums ja SVM saavutasid tugevalt

statisiliselt olulised tulemused, vastavalt 38% ja 37% (juhusliku ennustuse täpsus oleks olnud umbes 11%). Mõlema probleemülesande puhul kasutasime me ka lähima naabri algoritmi, aga selle täpsus oli võrreldes eelmainitud meetoditega märgatavalt väiksem. Kuna roti asukoht on pidev muutuja, siis me proovisime käsitleda seda ka pideva ennustuse probleemina. Suurem osa regressiooni algoritme, mida selles töös analüüsitakse (kantregressioon (ridge regression), lassoregressioon (lasso regression), elastne võrk (elastic net)), andsid juhuslikule ennustustäpsusele lähedasi tulemusi. Ainult juhumeets andis pideva ennustuse probleemi puhul teistest meetoditest oluliselt parema täpsuse.

Seejärel analüüsisime me andmeid, mis olid salvestatud eksperimendist, kus rotid olid treenitud valima vasakut või paremat suunda number 8 kujulises labüündis, olles samal ajal ise jooksurattal. Nende mõõtmistulemuste puhul teostasime me esimese sammuna andmete mõõdete vähendamise (dimensionality reduction), et visualiseerida muutusi andmetes otsuse langetamise hetkel. Muuhulgas identifitseerisime ja tõime joonistel välja ka episoodirakud - neuronid, mis on rohkem aktiivsed kindlal ajal antud ülesande jooksul. Episoodirakud võivad kaasa aidata aja tajumisel ja episoodilise mälu loomisel. Samuti visualiseerisime neuronaalseid trajektoore otsuse langetamise ajal, et ette aimata, millise otsuse loom vastu võtab.

Kokkuvõtteks andis roti asukoha ennustamisel algoritmidest täpseimaid tulemusi juhumeets. See võib muuhulgas näidata seda, et informatsioon roti asukoha kohta sisaldub mitte-lineaarses neuraaalses aktiivsuses, mida lineaarregressiooni meetodid ei olnud võimelised tuvastama. Edasises uurimistöös plaanime me dekodeerida roti asukohta, kasutades meetodeid, mis on sarnasemad aju enda mehhanismidele. Neurovõrgud (neural networks) on laialt levinud masinõppe meetod, mis sarnaselt juhumeetsadega suudab ära tunda mitte-lineaarseid mustreid. Selles töös loodud andmetöötluskonveiereid (data processing pipeline), mis tegelevad üsnagi keerulise andmete eeltöötluste, tunnuste eraldamise ja andmestiku visualiseerimisega, panevad tulevikuks tugeva aluse hipokampuse dünaamika uurimisele TÜ arvutusliku neuroteaduse töögrupis.

Võtmesõnad: hipokampus, ajuandmete analüüs, masinõpe, navigeerimissüsteem, asukoharakud, dekodeerimine

Contents

Introduction	6
Methods	9
2.1 Where the data comes from	9
2.1.1 Experiment 1 - Exploring the environment	9
2.1.2 Experiment 2 - left/right alternation task	11
2.2 Data structure	12
2.3 Used algorithms	12
2.3.1 Supervised Learning	12
2.3.2 Unsupervised Learning	17
2.4 Measures of the decoding performance	20
Results	22
3.1 Navigation in space	22
3.1.1 Tests to identify place cells	22
3.1.2 Location - prediction task	24
3.1.3 Comparison of linear and Non-linear methods	34
3.2 Navigation in time	38
3.2.1 Neural dynamics during delayed alternation task	38
3.2.2 Time cells	39
3.2.3 Visualization of neuronal trajectory	41
Discussion	44
4.1 Summary of Results	44
4.2 Literature comparison	45
4.3 Critics	45
Conclusion & Future	48

Introduction

Navigating efficiently in the environment is one of the most crucial functions of our brain. The survival of many animals depends on their ability to navigate in a changing environment in order to quickly find their nest or food locations. This sense of place helps us for example understand where the nearest bathroom from your desk is at this moment. Neuroscientists have shown that the brain creates internal map-like representation of a place which allows animals to have an incredible sense of direction and provides them with ability to find a possible optimal path for the desired destination.

Lesion and imaging studies indicate that the main region of the brain responsible for our navigation and memory abilities is the hippocampus. The hippocampus contains “place cells” which fire when an animal reaches one specific place in the environment but stay inactive in other times. Recently, “grid cells” were found next to the hippocampus, that also fire depending on a certain point in the environment yet these points are arranged in hexagonal pattern. The combination of these two types of neurons creates an internal animal GPS system. For this discovery, the 2014 Nobel Prize in Physiology was awarded to Dr. John M. O’Keefe, Dr. May-Britt Moser and Dr. Edvard I. Moser.

The exploration of this part of the brain and more precisely the exploration of these types of neurons, gives us information on how we process the external world and how spatial navigation and temporal navigation (memory) are related. For example, it is known that “place cells” not only depict the current location but also contain information about where the subject just has been and where it is planning to go. Therefore, it is thought that recalling a memory is closely related to deciphering the place and context experienced in the past and planned in the future.

To better understand the connection between the neuronal activity and spatial navigation, in this project we will apply different machine learning algorithms to high-quality data from rats hippocampus. In particular, we will inspect data collected by Prof. Buzsaki from the NYU, head of one of the most important labs in rodents brain research. The experiments were recorded with state of the art technology and contain the data obtained from thousands of rat neurons while these rats moved freely in a maze and performed different tasks.

The scientific goal of this thesis is to visualize and explore the high-dimensional patterns of neuronal activity in the hippocampus and how this relates to the rat navigation. Specifically, we will compare the power of different linear and non-linear

classifiers to predict an animal's location only based on its neuronal activity. We also plan to use dimensionality reduction algorithms to provide with an approximate visualization of the neuronal trajectory while an animal makes decisions. Such visualizations in 2 and 3 dimensions are highly appreciated in neuroscience studies as they allow the formation of hypothesis about neuronal dynamics.

Current recording techniques can detect only few hundred neurons in a system that contain hundreds of thousands of cells. Therefore, it is remarkable that one can decode animal position just based on the animal's neural activity. To do so, it is interesting to see which algorithm provides with a better accuracy and performance from this highly subsampled data.

Background information about place cells

Place cells are group of neurons in hippocampus which give us the sense of location at all times but also provide us with the ability to understand how to reach our desired destination. Questions about how living organisms are able to navigate that well in their environment goes back a long time ago, but the idea that the brain creates field map representations was proposed by the American psychologist Edward Tolman [1]. Although, the suggestion was opposed by other behaviourists, John O'keefe together with Dostrovsky (1971) found place cells from recordings of neural activities of the rats' hippocampus while the rats were moving around in an experimental area. They reported that these cells fired when an animal was reaching one particular place but stayed inactive when the animal was elsewhere.

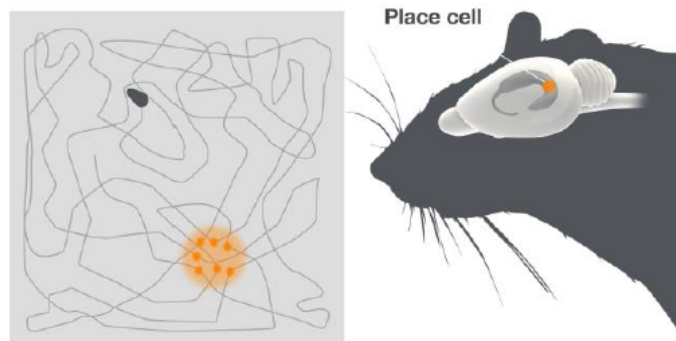


Figure 1.1: Example of place cells. grey area indicates experimental arena where a rat is moving around and yellow dots show its location where particular neurons fire. Figure from *The Brain's Navigational Place and Grid Cell System*[1]

From this discovery with some hippocampal-lesion studies, O'Keefe Nadel (1978) introduce a new idea that the hippocampus creates cognitive maps and that the place cells are the basic units of these maps [2]. This theory proposed the idea that each place cell represents one specific place in the environment and a combination of place

cells creates internal map representations which gives awareness to an animal about its position relative to other essential locations. Furthermore, this theory suggests that hippocampal lesions cause learning, memory and performance problems on the task where spatial cognitive processing is involved and have no impact on tasks where no orientation skills are required [2]. In spite the fact that many neuroscientists agree that the hippocampus has a crucial role in learning and memory, it is still unknown what is its precise role these processes.

What is known is that it generates internal maps as mentioned and also plays critical role in forming specific form of memory, called episodic memory [3] - the ability to remember specific events in time.

Methods

2.1 Where the data comes from

The data was obtained multi-neuron electrophysiological data recorded from the Buzsaki lab which offers numerous data sets collected from various experiments involving many brain areas. For this thesis, we analyse hc-3 and hc-5 data sets which contain data recorded from different experiments.

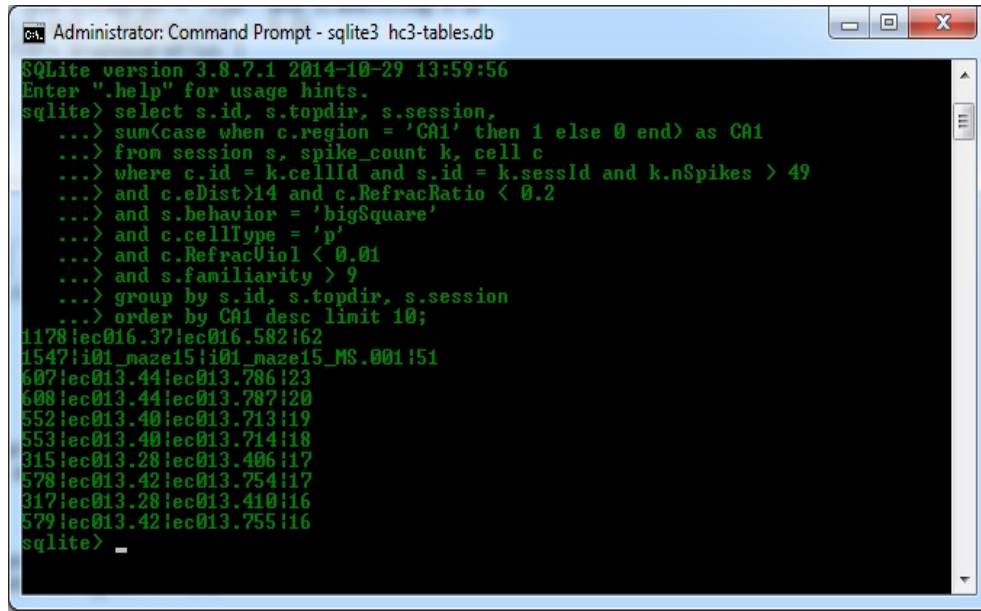
2.1.1 Experiment 1 - Exploring the environment

Hc-3 data sets contain recordings from experiments where 11 rats performed multiple behavioral tasks. To record the data, electrodes (shanks) were implanted in the hippocampus or the entorhinal cortex of the animals. Two LEDs were also attached on the animal's head in order to identify the coordinates of the animal while doing a task. Test subjects were completing one of 14 behavioral tasks during a session and several of them during a day. There were 442 sessions during which 7736 cells were detected in total[4].

In addition, the lab provides metadata files which include information on how data set is stored in different tables. One of these metadata files is hc3-tables.db which contains SQLite database tables. By using database queries and Windows command prompt, the data set can be filtered to find sessions and neurons of interest.

The figure 2.2 provides one example on how a desired session was extracted. In this case, the query gives 10 top sessions that contain most cells from 'CA1' region. The filtering is done also in terms of quality of the recorded cells, the performed behavior task, the cell type and familiarity. The query is shown below:

```
select s.id, s.topdir, s.session,
sum(case when c.region = 'CA1' then 1 else 0 end) as CA1,
from session s, spike_count k, cell c
where c.id = k.cellId and s.id = k.sessId and k.nSpikes >49
and c.eDist >14 and c.RefracRatio <0.2
and s.behavior = 'bigSquare'
and c.cellType = 'p'
```



```
Administrator: Command Prompt - sqlite3 hc3-tables.db
SQLite version 3.8.7.1 2014-10-29 13:59:56
Enter ".help" for usage hints.
sqlite> select s.id, s.topdir, s.session,
...> sum(case when c.region = 'CA1' then 1 else 0 end) as CA1
...> from session s, spike_count k, cell c
...> where c.id = k.cellId and s.id = k.sessId and k.nSpikes > 49
...> and c.eDist>14 and c.RefracRatio < 0.2
...> and s.behavior = 'bigSquare'
...> and c.celltype = 'p'
...> and c.RefracViol < 0.01
...> and s.familiarity > 9
...> group by s.id, s.topdir, s.session
...> order by CA1 desc limit 10;
1178!ec016.37!ec016.582!62
1547!i01_maze15!i01_maze15_MS.001!51
607!ec013.44!ec013.786!23
608!ec013.44!ec013.787!20
552!ec013.40!ec013.713!19
553!ec013.40!ec013.714!18
315!ec013.28!ec013.406!17
578!ec013.42!ec013.754!17
317!ec013.28!ec013.410!16
579!ec013.42!ec013.755!16
sqlite> _
```

Figure 2.2: Filtering the data set by using SQLite

and c.RefracViol < 0.01
and s.familiarity > 9
group by s.id, s.topdir, s.session
order by s.session desc limit 10;

We used different criteria to obtain a session that contains relevant neurons. For example:

- eDist - "Isolation distance" (Harris, Hirase, Leinekugel, Henze and Buzsaki, Neuron, 2001).
- RefracRatio - an interspike interval index "R2/10" (Fee, Mitra and Kleinfeld, Journal of Neuroscience Methods, 1996)
- RefracViol - fraction of interspike intervals less than 2 milliseconds (Takehara-Nishiuchi and McNaughton, 2008)
- Behavior - performed behavioral task by the animal
- celltype - whether the neuron of interests is pyramidal or interneuron
- familiarity - number of times a task has been performed by animal

Detailed information can be found in [crcns-hc3-data-description.pdf](#) and [crcns-hc3-processing-flowchart.pdf](#) files in [www.crcns.org](#)

2.1.2 Experiment 2 - left/right alternation task

Hc-5 data set consists of recordings from three brain regions (left and right hippocampal CA1 and right entorhinal cortex (EC)) of a rat while the animal was performing various behavioral tasks. In this experiment, the tasks include a left/right alternate selection task, wheel running, and platform exploration. Rats were trained to enter in the wheel after each maze run and stay there for some time[5]. Figure 2.3 represents maze and the wheel.

The data includes:

- Waveforms of putative spikes extracted from original raw broadband signal
- LFPs (local field potentials)
- Results of spike sorting
- Information about the animal behavior during the experiment[5]

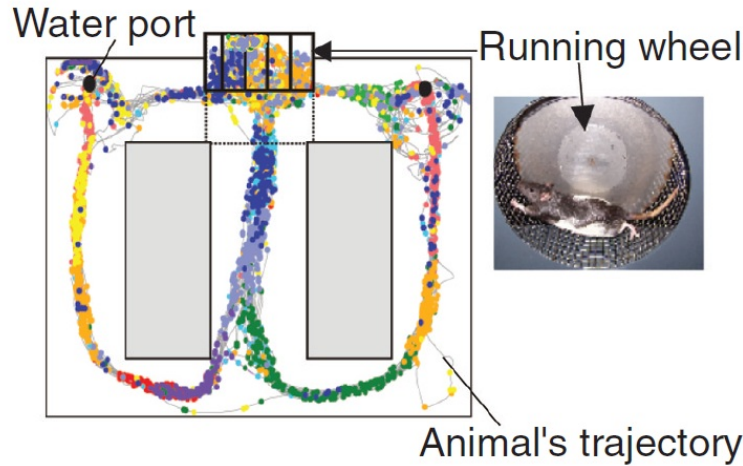


Figure 2.3: The task of the rat was to run in the wheel after each maze run and choose left or right direction afterwards. While running in the wheel, the animal was facing to the left. Colored dots are CA1 pyramidal neurons also called place cells [6].

Full description of the data could be found in [crcns-hc5-data-description.pdf](#) file from [www.crcns.org](#) website.

2.2 Data structure

In order to extract a usable matrix from hc-3 data, for machine learning purposes, the following actions have been taken:

- Since video recording (39.06 Hz) and spike detecting (20 kHz) have different sampling rates, we converted these two rates in milliseconds to correspond to one another.
- All clusters (neurons) detected by every shanks were put in one single file with its corresponding time in milliseconds.
- Our final matrix, which could be used for machine learning purposes, has been extracted by counting how many times each neuron fired every 200 milliseconds from the beginning of the session.
- As a result, the features of the matrix correspond to the neurons detected during the session and each row depicts a 200 milliseconds time frame.

For the data corresponding to the left/right alternation task, we followed a similar procedure as described above for the hc-3 data. However, to count each neuron's activity in a time window (estimation of a time-varying rate), we convolved each spike train with Gaussian kernel. The purpose was to find relationships between neurons firing rates during a wheel running and the following decisions.

2.3 Used algorithms

In this thesis, we use several machine learning algorithms for different purposes. In the first part, we apply supervised learning techniques to classify an animal's location and for the second part, we used unsupervised learning algorithm in order to reduce the data dimensionality and to visualize the results.

2.3.1 Supervised Learning

In general, supervised learning algorithms are implemented for classification problems. An algorithm builds a model or learns from input training data. Based on the model, it is supposed to label yet unknown object as one of a known group of objects from the training data. Afterwards, a testing data is used to validate how well the algorithm performed before it is applied on some new data. Performance of a classifier is usually compared to a random chance level in order to have a better understanding of the results. One way to calculate a random chance level is to calculate accuracy of the classifier which randomly predicts equally distributed classes. For example, if we had data containing two classes of objects, accuracy of randomly classified data would equal to around 50%. Another way is to calculate accuracy of the classifier which

predicts only one class that is the most common in a data set. Since in our case, we do not have equally distributed classes (unbalanced set), we consider the second way more relevant to produce a random chance level.

We used different linear and non-linear supervised learning algorithms to predict a rat's position in the experimental arena.

2.3.1.1 Linear Methods

Linear models try to find the best straight line to put on the data points and model dependency among variables or separate into two classes of objects depending on whether it is a regression or classification task. For Regression analysis, we used Ridge regression, LASSO, and Elastic Net. For classification, only Support Vector Machine has been used as linear method but it also can be used as non-linear method when some non-linear kernel is applied.

Least Mean Squares (LMS). Least mean square algorithm is one of the most known techniques to find the best straight line for a data set. It tries to minimize the sum of the mean squared errors between actual and predicted values given by a model. To do so, the algorithm starts defining coefficients randomly and step by step converts them until the errors are minimized.

Suppose, we need to choose the best fitted $y = \theta_1 + \theta_2 x$ line for a given dataset $\{(x_1, y_1); (x_2, y_2); \dots (x_m, y_m)\}$. In order to obtain the most suitable coefficients (θ_i) for the data, the first step is to define Cost or Loss function as follows:

$$Loss = \frac{1}{m} \sum_{i=1}^m (\theta_i x_i - y_i)^2 \quad (2.1)$$

Where $x_1 = 1$. To minimize this function, the Gradient Descent algorithm can be used because it initializes its search of the best line with random coefficients. During each iteration it updates them in order to receive each time lower error. Each iteration is calculated by using partial derivatives with respect to θ_i :

$$\theta_i = \theta_i - \alpha \frac{\partial}{\partial \theta_i} \frac{1}{m} \sum_{i=1}^m (\theta_i x_i - y_i)^2 \quad (2.2)$$

where α is a learning rate. If α is too small, then the algorithm would need more time to optimize the function. However, if it is too big, then the function might not be optimized at all.

For this thesis, algorithms will be applied to multidimensional data where Y is an m by 1 vector of responses, X is m by n matrix of predictor features and β is n by 1 vector of unknown coefficients. Accordingly, Loss function may be written in the following ways:

$$Loss = (\theta X - Y)^T(\theta X - Y) \quad (2.3)$$

Ridge Regression. If a model is complex, in other words, has too many features, fitted model may work pretty well on a training data but would fail to make predictions for some new data. This problem is often referred as overfitting.

To solve this problem, Ridge Regression algorithm penalizes the square size of the model coefficients. By adding $\sum_{j=1}^n \theta_j^2$ to the equation (2.1), we get a regularized version of the Least Mean Squares algorithm known as Ridge Regression.

$$\frac{1}{m} \sum_{i=1}^m (\theta_i x_i - y_i)^2 + \lambda \sum_{j=1}^n \theta_j^2 \quad (2.4)$$

λ is called a regularization parameter. As it approaches to plus infinity, the coefficients are getting closer to zero which may cause a problem of underfitting. Yet, if λ is too small, then it has no effect on the algorithm and consequently, the results are similar to the LMS algorithm.

Least Absolute Shrinkage and Selection Operator (LASSO). LASSO algorithm also penalizes the size of the regression coefficients but instead of using squared penalty as Ridge Regression does (2.4), it uses $\sum_{j=1}^n |\theta_j|$ penalty:

$$\frac{1}{m} \sum_{i=1}^m (\theta_i x_i - y_i)^2 + \lambda \sum_{j=1}^n |\theta_j| \quad (2.5)$$

Although, these two algorithms(2.4 and 2.5) may seem similar to each other, they output completely different results. By applying not squared penalty, some coefficients will be ultimately shrunk down to zeros which means that the algorithm selects more useful variables. In case of Ridge Regression, however, all variables are used for prediction.

Elastic Net. Elastic Net implements the combination of penalties of LASSO and Ridge Regression. As a result, the algorithm not only builds a sparse model where some predictors are non-zero like LASSO does but also it has the regularization properties of Ridge Regression.

$$\frac{1}{m} \sum_{i=1}^m (\theta_i x_i - y_i)^2 + \lambda_1 \sum_{j=1}^n |\theta_j| + \lambda_2 \sum_{j=1}^n \theta_j^2 \quad (2.6)$$

Support Vector Machine. Support Vector Machines (SVM) is another powerful supervised learning algorithm in the machine learning field. The key point of SVM is to find the hyperplane which gives the largest margin, in other words, the largest distance between two different classes of training examples. As explained in *Introduction to Pattern Recognition* [11], in order to get familiar with the term of margin, we should consider the linear classifier:

$$w^T x + w_0 = 0 \quad (2.7)$$

The maximum margin we seek is the area between these two parallel lines

$$w^T x + w_0 = 1, w^T x + w_0 = -1 \quad (2.8)$$

2.3.1.2 Non-Linear Methods

In many cases, data is not linearly separable, that is to say, there is no line that can separate two classes of objects in space features. At this time, we need algorithms that deal with classification problems in a different way rather than trying to use the approach of linear methods. One example of a non-linear approach is to construct a top-down decision tree in which nodes are points where choices are made and branches represent possible alternatives. Figure 2.5 shows one simple decision tree to help financial officers whether they should give a loan to a customer or not.

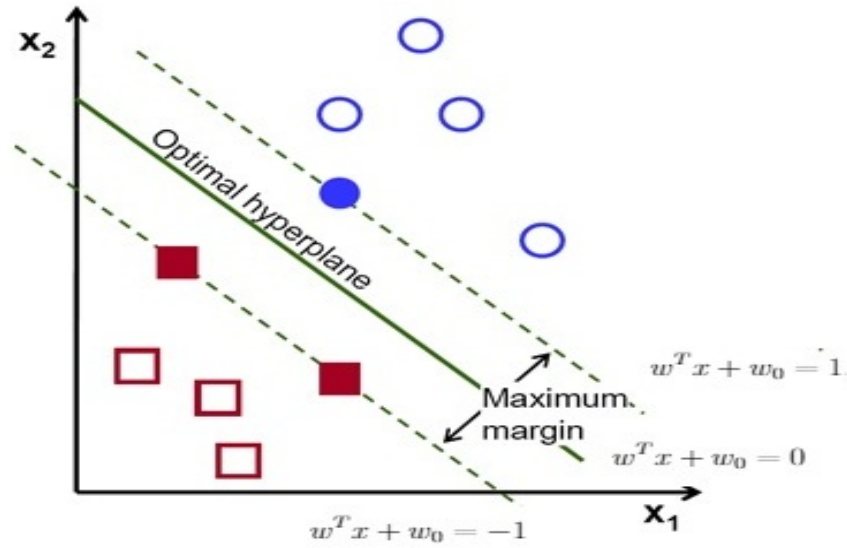


Figure 2.4: Simple example of how SVM classifies data. (Image source from url: http://docs.opencv.org/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html)

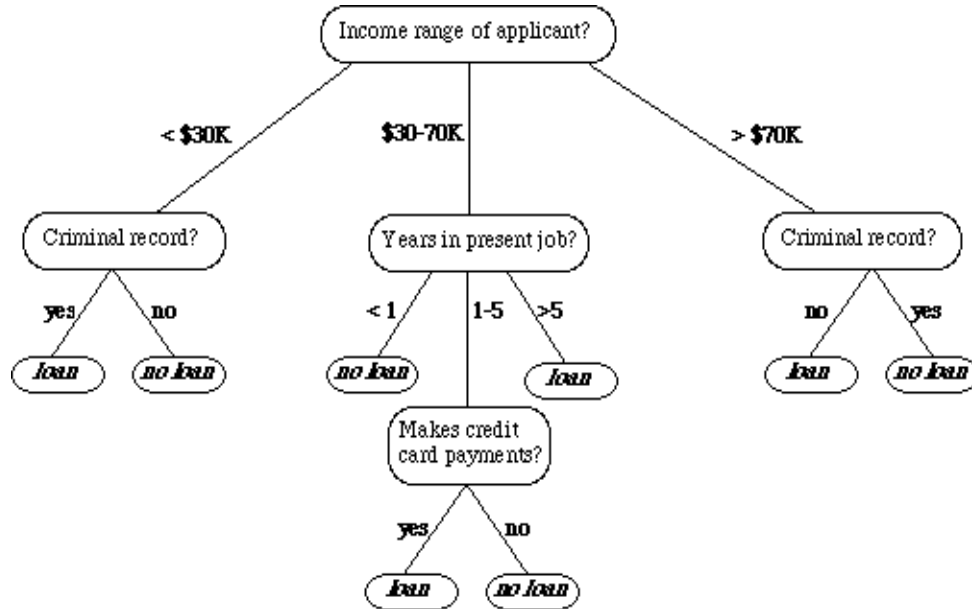


Figure 2.5: Simple example of a decision tree for financial decisions. The figure is from website: [www.cse.unsw.edu.au](http://www.cse.unsw.edu.au/billw/cs9414/notes/ml/06prop/id3/id3.html) (link: <http://www.cse.unsw.edu.au/billw/cs9414/notes/ml/06prop/id3/id3.html>)

In this thesis, Random Forest and K-nearest Neighbor have been used as non-linear methods to classify the data.

Random Forests. Random Forests is one of the most powerful machine learning techniques which can be used for both regression and classification problems. The algorithm is a collection of a large number of decision trees built based on a training data set. As described in *Classification and Regression by randomForest* [7] every classification or regression trees are constructed from N number of independently bootstrap samples and instead of choosing the best split for each node of each tree (as it happens for normal classification/regression tree), it chooses the best split from M randomly selected predictors. Random Forests makes predictions for new data by counting the majority votes from N trees for classification or take the average for regression problems. The algorithm has many advantages, like for example, it is very easy to set parameters and can deal with missing data. Besides these facts, it should be also mentioned that Random Forests cannot predict completely new values which are not in range of a training data set when it is applied for regression tasks.

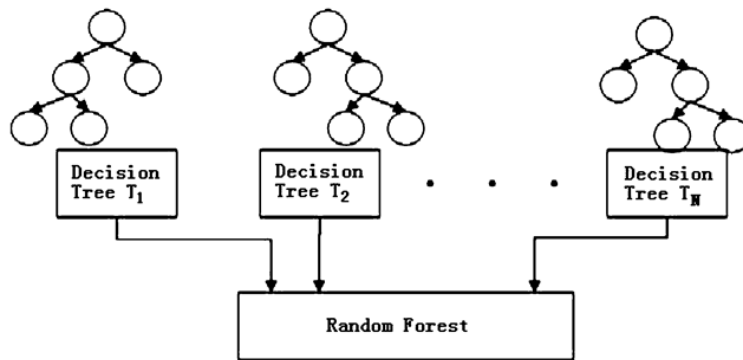


Figure 2.6: Illustration of Random Forests. Image from <http://pixgood.com/random-forest.html>

K-Nearest Neighbor (KNN). KNN is another method for classifying data which is very simple to understand but works very well in practice. KNN is considered as lazy algorithm since there is no explicit training phase or it is very minimal[12]. It keeps the training data for the testing stage and an object is classified by counting majority votes from its neighbors.

KNN puts data in feature space and calculates distance between an object which should be classified and other known objects from training data. If $k = 1$, then the algorithm labels the object according to the nearest object's label.

2.3.2 Unsupervised Learning

As opposed to supervised learning, unsupervised learning tries to build its model based on unlabeled data. By implementing these kinds of algorithms, one could cluster objects in a group in a way that objects in a same group would have similar

characteristics. Unsupervised learning algorithms are also used to reduce dimensionality of very heterogeneous data. In particular for this thesis, we applied unsupervised learning in order to reduce dimensionality of neuronal data. For this purpose, we used principal component analyses.

2.3.2.1 Principal Component Analysis

Principal Component analysis (PCA) is a very useful statistical tool from linear algebra to analyse and visualize complex data sets. It applies mathematical techniques to reduce dimensionality and underline strong patterns with minimal loss of information. In particular, PCA aims to find new a set of variables for the data, fewer than initial ones, which could describe most of the information (variance) the data includes.

To understand how PCA works, firstly one needs to get familiar some mathematical concepts such as covariance matrix, eigenvectors, and eigenvalues. Covariance measures how two variables change each other but it does so only between two dimensions[9]:

$$Cov(X, Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{n - 1} \quad (2.9)$$

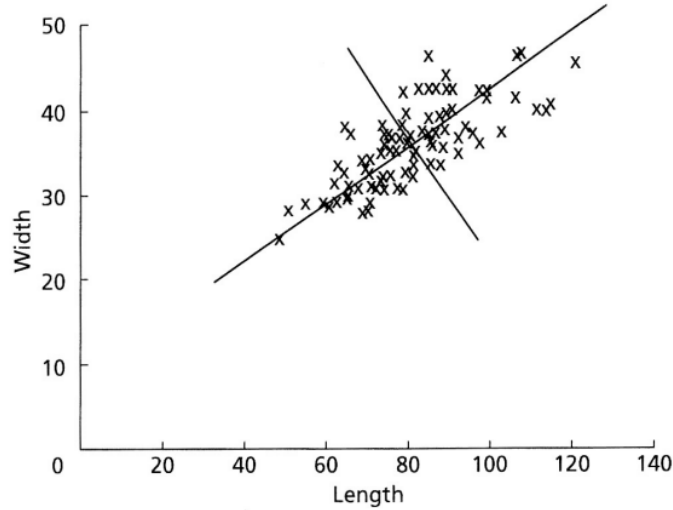
If a data contains more dimensions, calculating covariances between every two possible dimensions produces symmetric covariance matrix; where on the main diagonal there are covariance values between a variable and itself [9]. These values are also called variances. If we had 3 dimensional data, covariance matrix would like this:

$$\begin{pmatrix} Var(X) & Cov(X, Y) & Cov(X, Z) \\ Cov(Y, X) & Var(Y) & Cov(Y, Z) \\ Cov(Z, X) & Var(Z, Y) & Var(Z) \end{pmatrix}$$

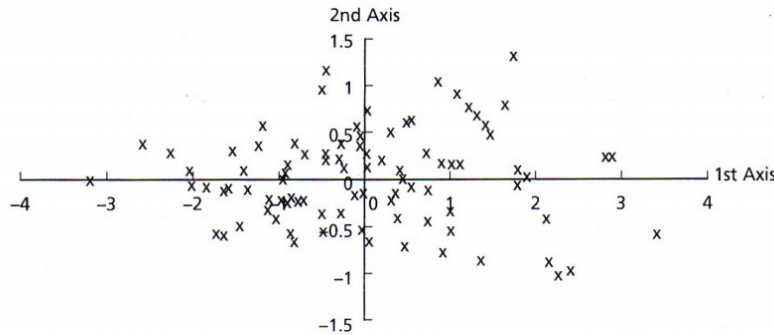
After obtaining a covariance matrix from n dimensional data, next step is to calculate eigenvectors and eigenvalues for the matrix. If we multiply our matrix A by v vector and resulting vector will be equivalent to v multiply by scalar value λ:

$$Av = v\lambda \quad (2.10)$$

we can call v as eigenvector and λ - eigenvalue. In general, $n \times n$ square matrix has only n number of eigenvectors and eigenvalues.



(a) find new axes for data set



(b) project data onto new axes

Figure 2.7: Transforming original data set onto new axes (eigenvectors) using PCA. (a) original data is represented with little crosses. Straight lines show found eigenvectors which are orthogonal to each other. (b) figure represents transformed data onto the new axes. These figures are from *Swan and Sandilands, 1995*[8].

The key point of eigenvectors is that they are orthogonal to each other. After finding them for a particular matrix, one could transform data in a way that these eigenvectors will be new axes instead of original ones (See Figure 2.7). Principal components expressing the main information of data set are chosen between these new axes. Eigenvector that has the highest corresponding eigenvalue is the main principal component.

2.4 Measures of the decoding performance

In machine learning field, accuracy, confusion matrix, recall, precision and F1 are most commonly used validation methods to assess how well some algorithms performed in a particular case.

Confusion matrix is a table which gives full information on how many times an algorithm correctly classified or misclassified the data. Columns of this table represents predicted classes by an algorithm while rows shows actual classes.

	Class 1	Class 2
Class 1	85%	15%
Class 2	10%	90%

Table 2.1: Example of a confusion matrix. Rows of the table represent known classes and columns show predicted classes by an algorithm

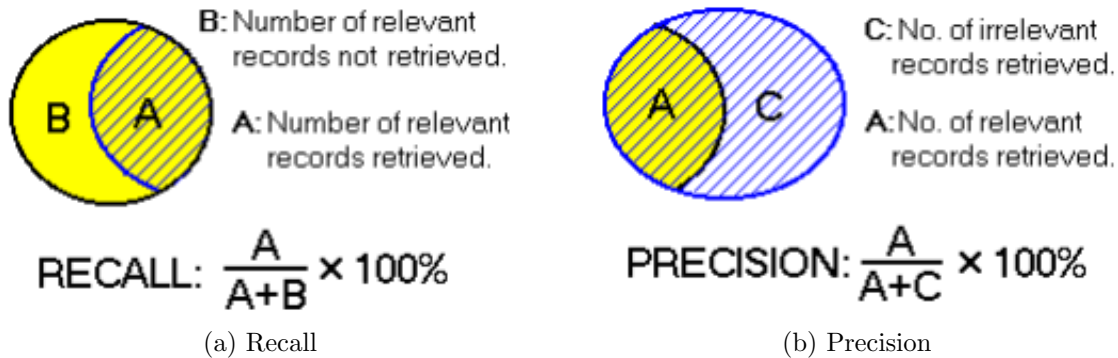


Figure 2.8: Formulas of calculating of Recall and Precision as illustrated with document retrieval problem. Both figures are from Creighton University website: www.creighton.edu (url: <https://www.creighton.edu/fileadmin/user/HSL/docs/ref/Searching-Recallprecision.pdf>)

If we had two classes of objects (data with two labels), then the table 2.1 would show that an algorithm classified Class 1 and Class 2 correctly in 85% and 90% of the cases respectively. It confuses Class 1 with Class 2 only in 15% and confuses Class 2 with Class 1 in 10% of the cases. In general, if the numbers on the diagonal of a confusion matrix are high, it means that given algorithm performed well.

Accuracy measures overall performance of an algorithm. It is calculated by dividing number of correctly predicted classes by total number of classes. However, sometimes calculating only Accuracy is not enough to understand how well algorithms classified objects. For more detailed information, it is useful to calculate Recall and Precision.

Recall measures how many times a correct class was found. For the example in Figure 2.8 (a), it is calculated as dividing number of relevant records retrieved by the total number of relevant records.

On the other hand, Precision is a measure of how many times a predicted class was correct. For the example in Figure 2.8 (b), It is measured as dividing number of relevant records retrieved by the total number of retrieved records.

Finally, F1 is used to determine accuracy from Recall and Precision. All together, they provide good characterization of performances of algorithms.

Results

3.1 Navigation in space

For the first part of the thesis, we predict a rat's location based on its neuronal activities which include so called place cells that fire only when an animal reaches particular place in the environment. We use several machine learning algorithms for this purpose and compare which one of them gives better results. Firstly, the rat's position was predicted into 4 blocks of the experimental arena, then into 16 blocks and afterwards, we tried to predict the exact location of the rat using linear regression algorithms.

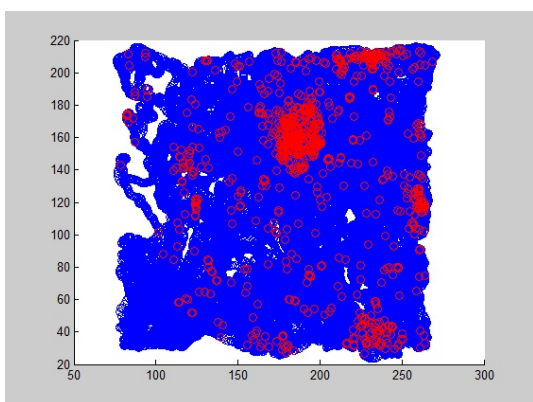
3.1.1 Tests to identify place cells

Before applying machine learning algorithms to the data set, it was necessary to make sure that some of the recorded neurons were actual place cells which are expected to have main role in decoding the rat's position.

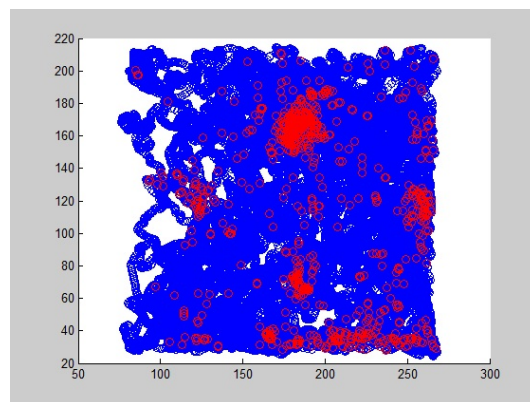
Since the data set provided information about when and at what place each particular neuron was active, we decided to compare neurons from two different recording sessions that had been conducted in the same day with the same animal. If a specific neuron was firing in only one particular place during one experiment and the same neuron was also active only in that place during the second experiment, we could conclude that this neuron was probably a place cell.

As mentioned above, Place cells are mostly located in one region of the hippocampus called "CA1" and they represent approximately 1/10 of total number of neurons in this area. Thus, by visualising neural activities only from this region of the brain, a relative number of place cells should have been detected.

After plotting each neuron's activities recorded in one session and comparing the same neuron's activities in the second session, it was clear that some neurons were firing in the same place of the arena during both sessions. There were several place cells detected and some of them can be seen below:

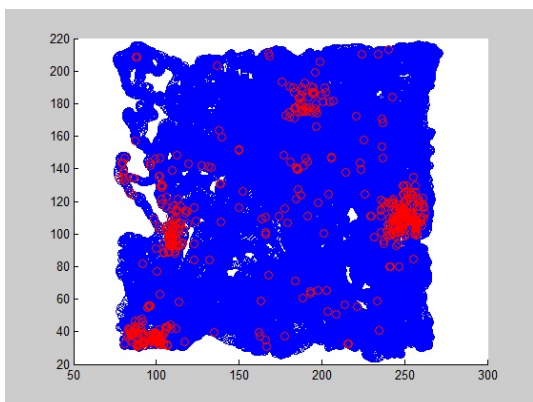


(a) Session "ec013.786"

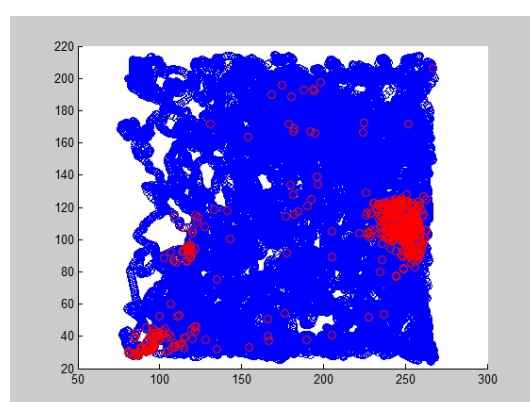


(b) Session "ec013.787"

Figure 3.9: Neuron #510

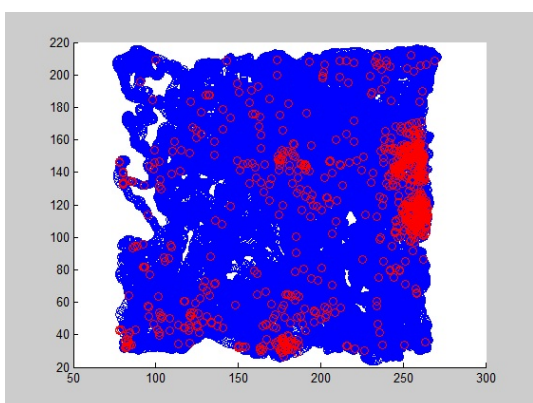


(a) Session "ec013.786"

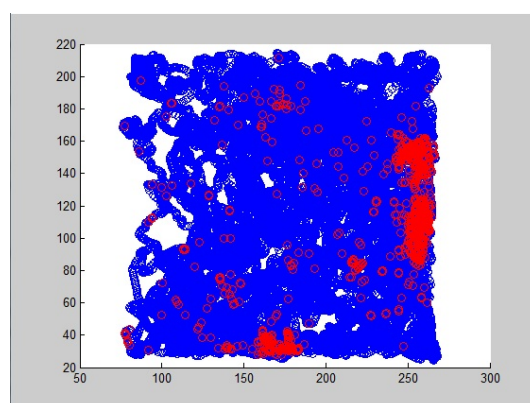


(b) Session "ec013.787"

Figure 3.10: Neuron #511



(a) Session "ec013.786"



(b) Session "ec013.787"

Figure 3.11: Neuron #605

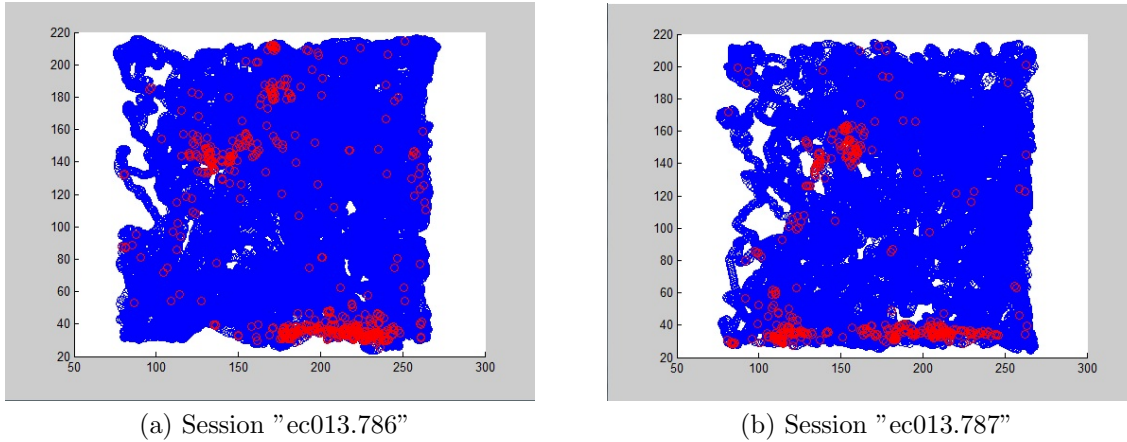


Figure 3.12: Neuron 807. This figure shows identified place cell which fires at the same place during the two experiments. These experiments were performed in a same day with the same experimental arena. Blue color in the figure shows covered area of the rat during the experiment and the red line displays at which place this particular neuron was firing. The same is true for Figures 3.9, 3.10, 3.11.

3.1.2 Location - prediction task

In general, when it comes to applying machine learning algorithms, data sets need to be divided into training and test data. The former is necessary to build and train a model, while the latter is used to measure how well a model works. To train a model, in other words, to learn from previous observations, machine learning algorithms must have inputs such as training data, corresponding response vector and other necessary parameters depending on an algorithm.

In this case, the first 66% of the whole data set was selected as the training data and the rest was used for validating results.

3.1.2.1 Classification

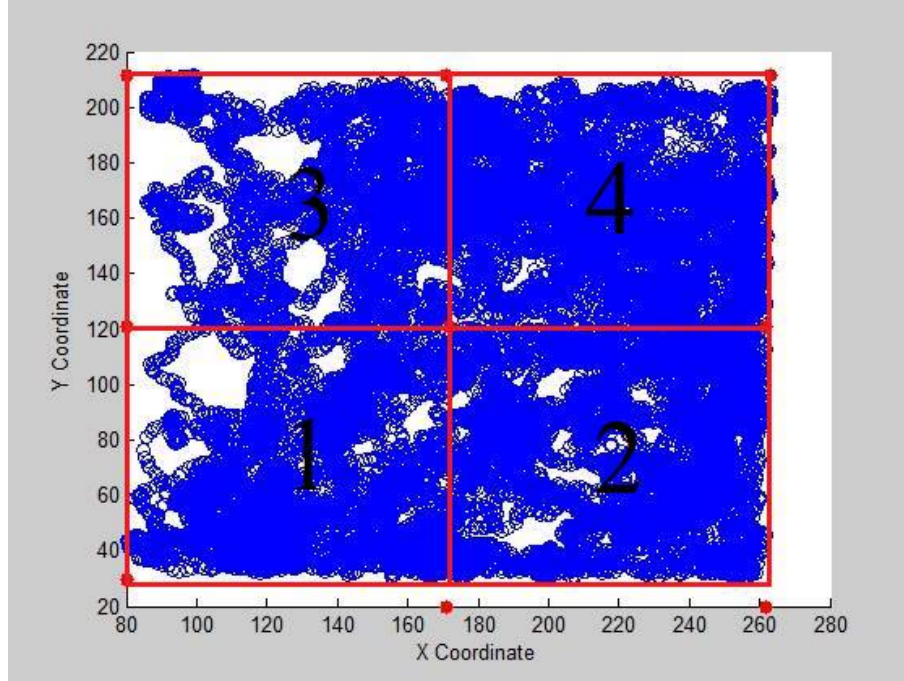


Figure 3.13: Experimental arena was divided into 4 blocks to apply classification algorithms, the blue color denotes the places where the rat has been in the arena during the experiment.

Classifying the rat's positions into 4 blocks

At first, we decided to divide the experimental arena into 4 blocks and apply classification algorithms to predict in which particular block the rat was in a given time based on its neural activity (See Figure 3.13).

Random Forest. As described in section 2.3.1.2, Random Forest builds classification trees from M randomly selected predictors. The most common and easy way of doing so in order to calculate this value is to take the square root of the total number of predictors. In this case, we have 61 predictor neurons and thus, M is around 7. The number of constructed trees we chose is 500. Although, we tried other values for these parameters, the best results were obtained with the mentioned numbers.

After the algorithm output the results, and in order to validate them, we had to find how many times the algorithm correctly classified and how many times it misclassified the objects. Thus, we calculated the confusion matrices. In order to visualize the performances, we also plotted the confusion matrix with colors and numbers of percentage for each class (See Figure 3.14). The columns of the plot are predicted

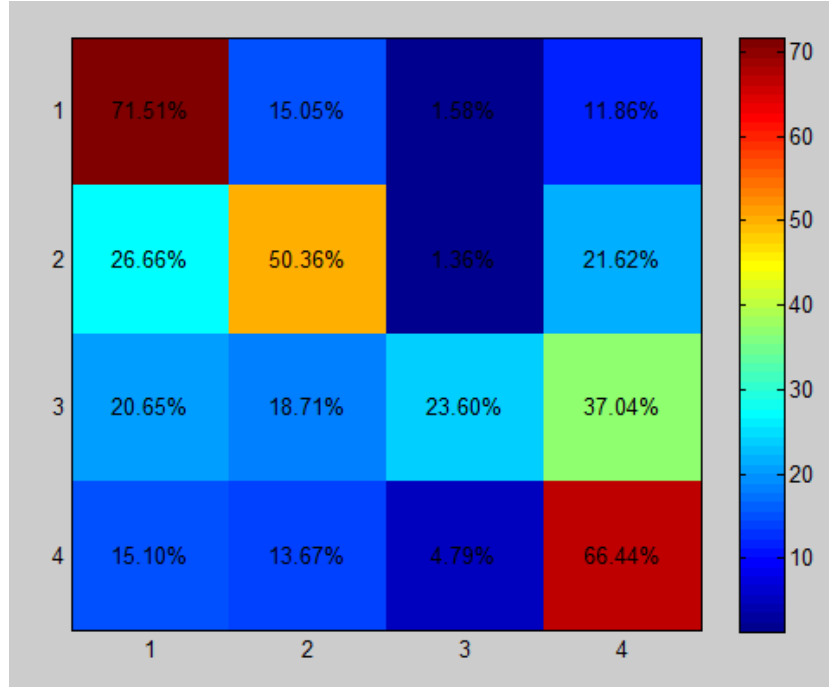


Figure 3.14: Confusion matrix based on performance of RF while classifying the rat's position in 4 blocks of the experimental arena.

classes and the rows represent the actual classes. The diagonal of the table indicates the percentage of correctly predicted values.

Considering the given confusion matrix, the algorithm correctly predicts when the rat is in block 1 in 71.51% of the cases. However, it confuses this block with block 2 in 15.05% of the cases, and respectively with block 3 and block 4 in 1.58% and 11.86% of the cases. Block 2 was correctly predicted in 50.36%, block number 3 - 23.60%, and block number 4 - 66.44% of cases. Since the rat was fewer times in block 3 (Figure 3.13), the prediction of this block is very poor and consequently, the algorithm confuses it with the other three blocks, mostly with block 4 (37.04%).

Because of the fact that we have four possible classes (blocks), randomly made predictions would give us approximately 31% accuracy for each class. Given the fact that accuracy of the algorithm for three classes is much higher, we can conclude that Random Forest was successful in making predictions about the rat's location using the neural data.

SVM. We used LIBSVM[10] tool to apply SVM algorithm to the data. It produced almost the same results as Random Forest with some interesting variations. The percentages of correctly classified block number 1, 2, and 4 are a bit less comparing to RF. However, SVM predicts block 3 more successfully (35.47%) than RF (23.60%).

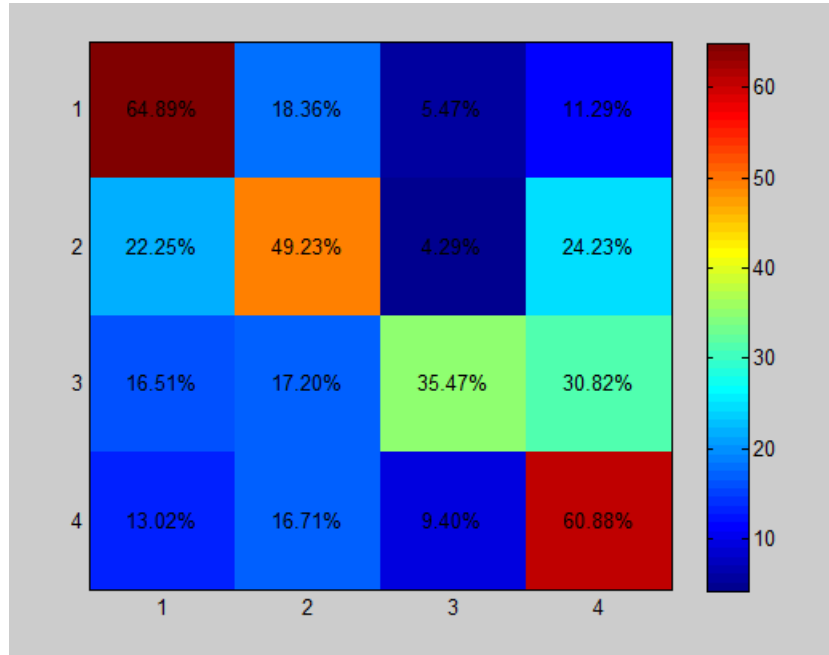


Figure 3.15: Confusion matrix based on performance of SVM while classifying the rat's position in 4 blocks of the experimental arena.

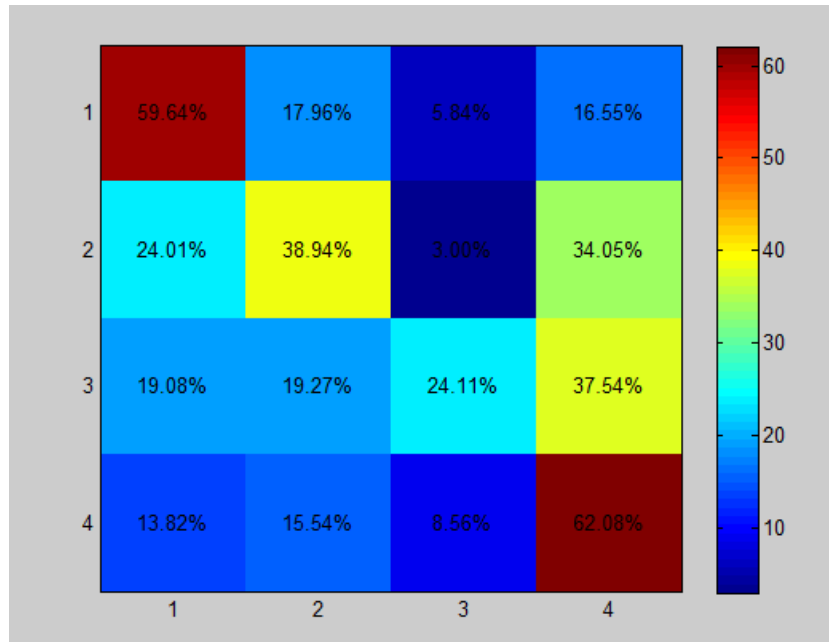


Figure 3.16: Confusion matrix based on performance of KNN while classifying the rat's position in 4 blocks of the experimental arena.

KNN. We tried several k values when applying KNN to the data and the best results were obtained with $k = 21$. However, this algorithm was the least successful for the

classification problem. It does the same mistake as RF, confuses block 3 with block 4 and furthermore, it also confuses block 2 with block 4 in 34% of cases which is the worst result regarding the given results for the same block. In general, we can say that SVM and RF were much better in almost every component.

Classifying the rats positions into 16 blocks

Since the algorithms have shown good results with the classification of the rat's location into four blocks, for the next step we made the classification problem even harder. Each block of the experimental area was divided into four blocks and we got 16 blocks in total. As a result, algorithms were supposed to predict in which of those 16 blocks the rat has been in a given time.

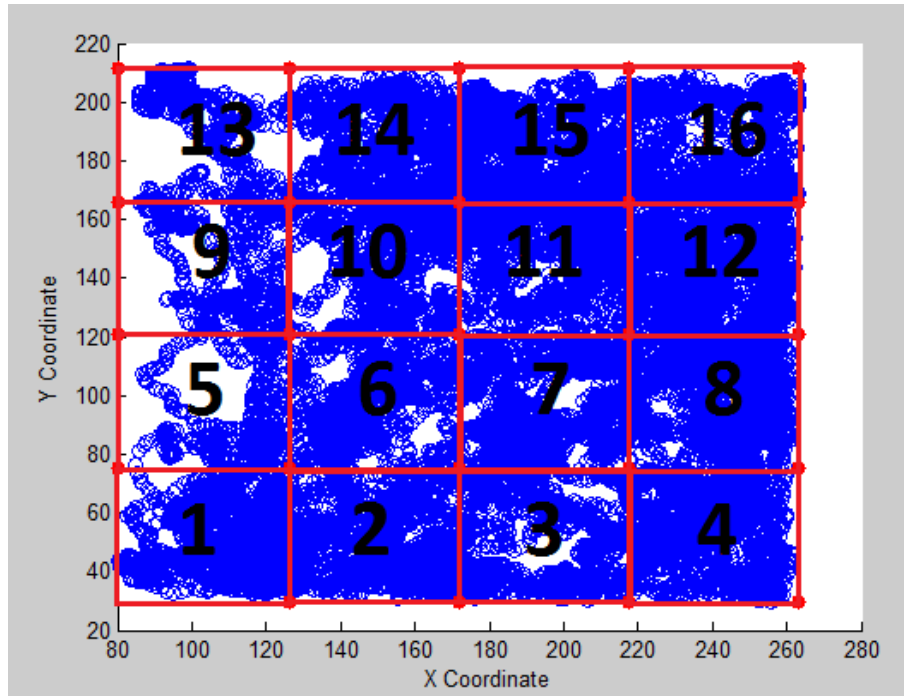


Figure 3.17: Experimental arena was divided into 16 blocks to apply classification algorithms, the blue color denotes the places where the rat has been in the arena during the experiment.

Random Forest. We used the same techniques to check how algorithms did as in the previous case. We calculated the confusion matrix and error rate. Figure 3.18 shows the confusion matrix for which the color bar on the right indicates how high the percentages are for each square.

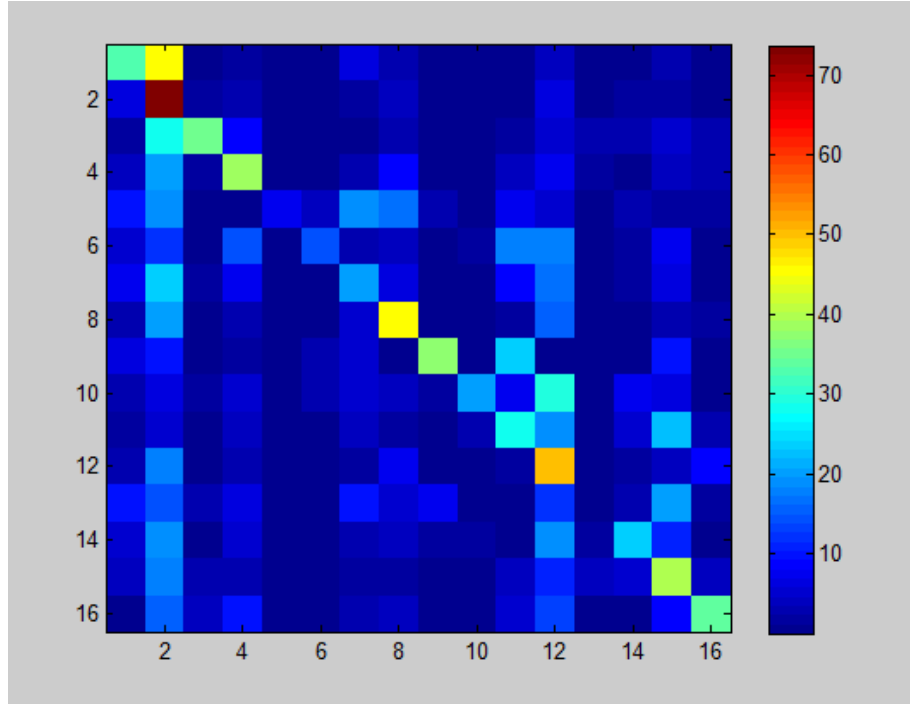


Figure 3.18: Confusion matrix based on performance of RF while classifying the rat's position in 16 blocks of the experimental arena.

Since the diagonal of the plot is distinguishable, it means that the algorithm performed well while dealing the classification task. However, this time the classification problem was four times more difficult as we increased number of blocks up to sixteen.

As the graph shows, in most cases the algorithm mistakenly classifies (46%) block 1 as block 2 which is logical because these two blocks are side by side. Also, in most cases it confuses block 13 with other blocks of the arena because the rat was fewer times in this block.

SVM. Although, the main diagonal on SVM confusion matrix might appear brighter than on the confusion matrix given by Random Forest, by looking at the color bar which represents percentage, we could see that in the case of Random Forest, the colors are adjusted in a way that the maximum number of correctly classified objects is 70%. In case of SVM, however, colors are adjusted according to 60%. This means that the same colors might represent different numbers in these two tables (Figures 3.18 and 3.19).

The number of correctly classified block 1, and the number that block 1 was misclassified as block 2 are the same according to the Figure 3.19. Just like Random Forest algorithm, SVM correctly predicts as well most of the times correctly when the rat is in block 2.

SVM also could not identify Block 13 in most cases and as a result, it was mistak-

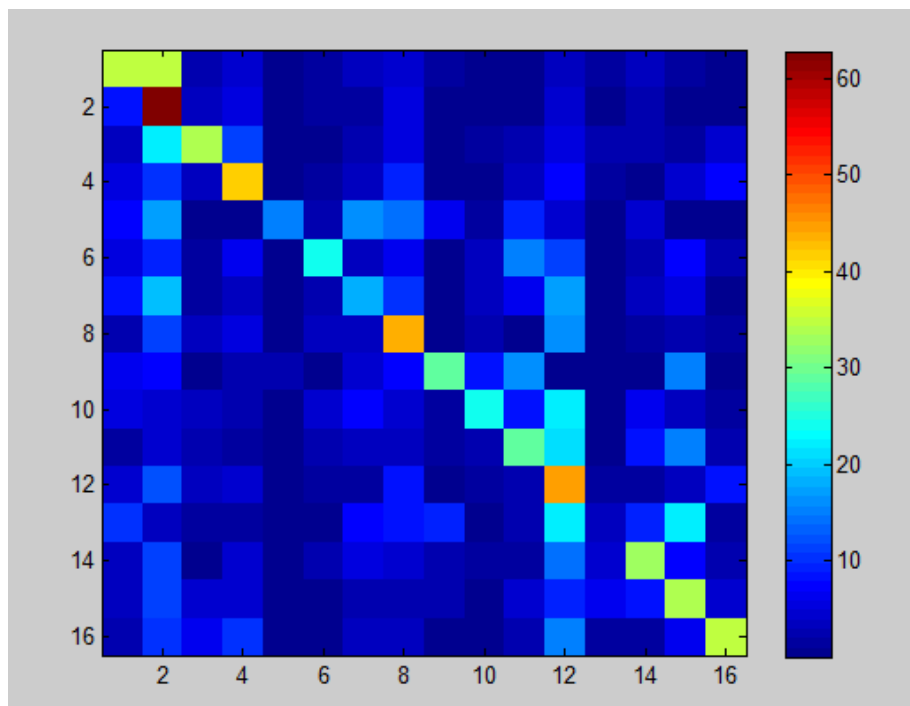


Figure 3.19: Confusion matrix based on performance of SVM while classifying the rat's position in 16 blocks of the experimental arena.

only classified as other blocks of the area.

KNN. KNN as in the previous case was least successful while dealing the classification problem. The diagonal of its visualised confusion matrix is less bright meaning that it made more mistakes than the other two algorithms. However, the structure of the confusion matrix seems more or less the same as the other two.

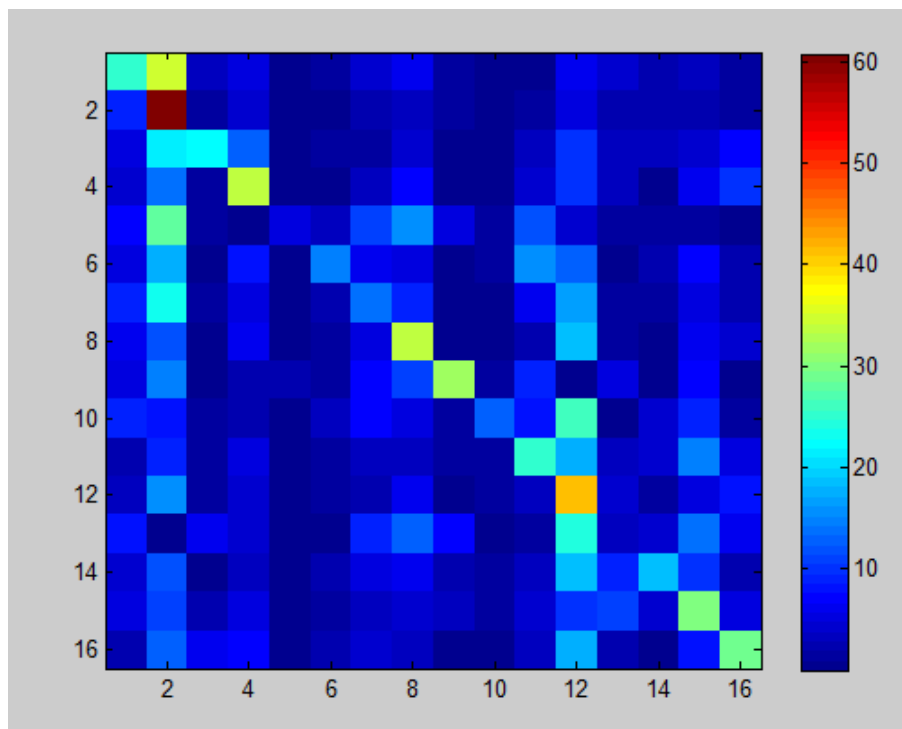


Figure 3.20: Confusion matrix based on performance of KNN while classifying the rat's position in 16 blocks of the experimental arena. Color bar represents number of percentage.

3.1.2.2 Regression

Since the rat position is a continuous variable, at last, we decided to predict the exact location of the rat in centimeters at a given time. To do so, linear/regression methods have been used to deal with the regression problem. We needed to predict both X and Y coordinates of the experimental arena. Firstly, a model was trained to produce possible X coordinates and then we followed the same process for Y coordinates. By the time we had both predicted vectors, they were easily combined into 2D location by finding their common time indexes and we could evaluate the performance of an algorithm.

Using linear/regression methods to predict the exact location of the rat at a time seemed a bit ambitious for these kinds of algorithms. However, they gave better results than chance MSE.

LASSO. To build the best model this algorithm can, the first step is to scan the λ parameter and make features selection accordingly. The figure 3.21 shows the process of LASSO algorithm while predicting X coordinates. The vertical axis of the graph represents MSE values and horizontal axis - scalar values of λ itself. The green circle,

on the left subfigure, shows value of λ for the model which minimizes the error, however, in order to avoid overfitting, it is proposed to choose the one with the blue circle which will cause mean square error to be around 1500 cm. The chosen value selects 53 parameters out of 61 (See Figure 3.21 (b)) and all others will be zeroed out. Figure 3.22 shows the same process but with Y coordinate.

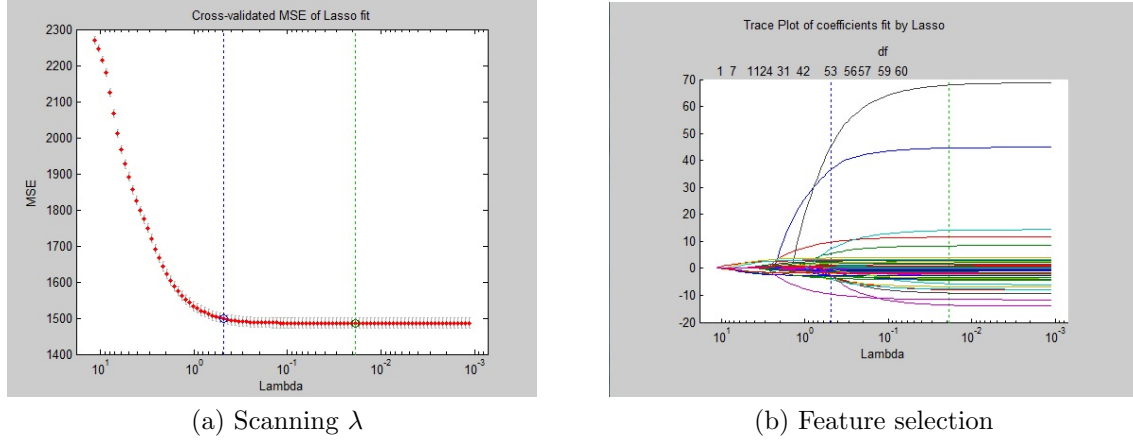


Figure 3.21: Finding the best penalty parameter λ to predict X coordinate. (a) shows Mean Square Error versus penalty parameter λ . The λ value which minimizes the error is indicated with the green circle but the value with blue circle is more recommended in order to avoid a problem of overfitting. (b) shows how many parameters are selected by each value of λ . In this case, λ with the blue circle selects 53 parameters out of 61.

In order to evaluate the performance of LASSO (See Figure 3.23) and the other algorithms, we decided to measure the Mean Error (ME) or Loss functions of these algorithms calculated by taking the square root of Formula 2.1 and compare them to chance Mean Error which is the sum of mean errors when the predicted value is always mean of a response vector. Given MEs by the algorithm were 39.61 cm and 45.28 cm while chance MEs were 45.46 cm and 54.28 cm for X and Y coordinates respectively.

Elastic Net & Ridge Regression. As described in section 2.3.1.1, LASSO, Elastic Net and Ridge Regression algorithms have very similar ways to train a model. Accordingly, we only report results of these algorithms because the process is already described in section 3.1.2.2.

ME produced by Elastic Net: 40.03 cm and 45.29 cm for X and Y coordinates respectively.

ME produced by Ridge Regression : 40.23 cm and 45.36 cm for X and Y coordinates respectively.

As the number shows, these three algorithms gave almost the same results but LASSO was a bit better.

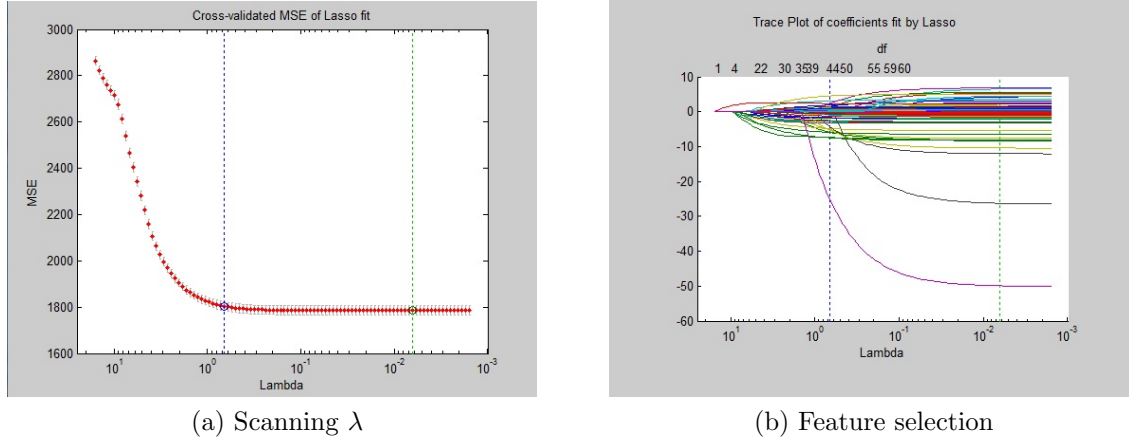


Figure 3.22: Finding the best penalty parameter λ to predict Y coordinate. (a) shows Mean Square Error versus penalty parameter λ . The λ value which minimizes the error is indicated with the green circle but the value with blue circle is more recommended in order to avoid a problem of overfitting. (b) shows how many parameters are selected by each value of λ . In this case, λ with the blue circle selects 44 parameters out of 61.

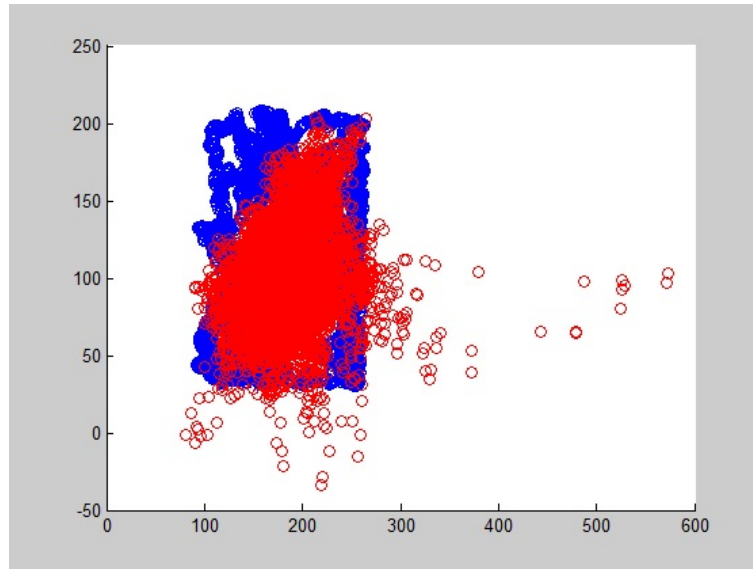


Figure 3.23: Predicted rat's positions based on its neural activity by LASSO algorithm: the rat's actual position is indicated with blue color, predicted location - with red color. Mean Errors (ME): For X coordinate 39.61 cm, For Y coordinate - 45.28 cm

Random Forest for the regression task. Random Forest can also deal with regression problems quite effectively. Its performance for this task was the best comparing to the linear methods. It gave the least ME in predicting both X and Y coordinates 35.62 cm and 40.03 cm for X and Y coordinates respectively (Figure 3.24).

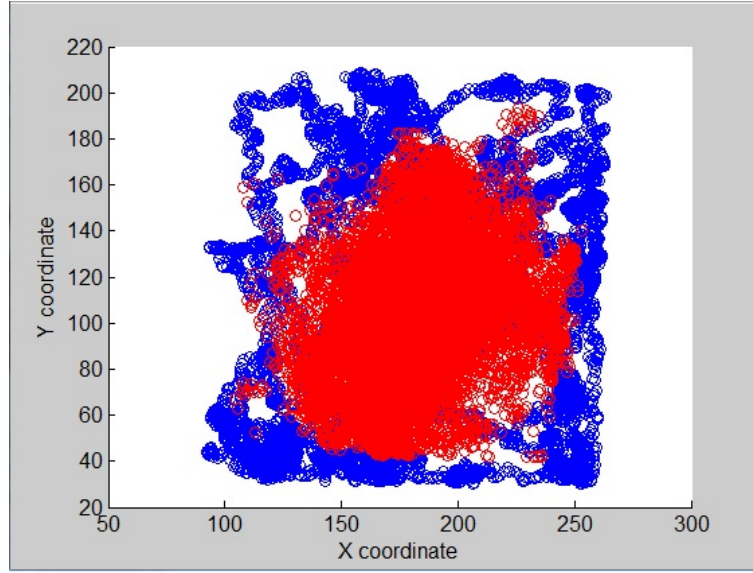


Figure 3.24: Predicted rat's positions based on its neural activity by Random Forest: the rat's actual position is indicated with blue color, predicted location - with red color.

3.1.3 Comparison of linear and Non-linear methods

We previously saw confusion matrices produced by different algorithms when classifying the rat's position into 4 blocks of arena. To better compare their performance, we also calculated the Accuracy of each algorithm to know how many instances their model classified correctly. This rate is calculated by dividing the number of correctly classified instances by the total number of instances. As a result, performances of the algorithms could be seen in Table 3.2

	Random Classifier	Random Forest	SVM	KNN
Accuracy	31%	57.8%	55.5%	49.6%

Table 3.2: Accuracy of the algorithms while predicting the rat's position into 4 blocks of the arena

Since Random classifier would give an accuracy of approximately 31%, the given algorithms performed much better and Random Forest was the best according to the results. But still those results do not give the full picture of how well they performed because, some of the classes must have been identified more easily and more often than others as we could see that Random Forest identifies block 1 71% of times. To obtain more details about their performances we calculated precision, recall and F1

for each block separately. Precision measures fraction of predicted classes that are correct while recall measures fraction of correct classes that are predicted. Recall is on the diagonal of the confusion matrices which we have seen in section 3.1.2.1. F1 is just a convenient way of a combining recall and precision as one number. Tables 3.3, 3.4, 3.5 show these numbers given by the algorithms.

	Block 1	Block 2	Block 3	Block 4
Precision	64.2%	55.6%	61.3%	51%
Recall	71.5%	50.4%	23.6%	66.4%
F1	67.68%	52.87%	34.09%	57.68%

Table 3.3: Precision, Recall, and F1 calculated from predictions made by Random Forest

	Block 1	Block 2	Block 3	Block 4
Precision	66.15%	51.60%	48.04%	49.35%
Recall	64.89%	49.23%	35.47%	60.88%
F1	65.51%	50.39%	40.81%	54.51%

Table 3.4: Precision, Recall, and F1 calculated from predictions made by SVM

	Block 1	Block 2	Block 3	Block 4
Precision	62.24%	46.01%	40.59%	42.08%
Recall	59.64%	38.94%	24.11%	62.08%
F1	60.91%	42.18%	30.25%	50.16%

Table 3.5: Precision, Recall, and F1 calculated from predictions made by KNN

In general, if these numbers are high, it indicates the better performance of these algorithms. According to the results, Random Forest and SVM gave almost the same results but Random Forest was a bit better in more cases than other way around. KNN, however, was quite behind comparing to both algorithms.

In order to reduce presentation complexity, we only show the accuracy of the algorithms from classification task into 16 blocks of the arena.

	Random Chance	Random Forest	SVM	KNN
Accuracy	11%	38.05%	37%	30.47%

Table 3.6: Accuracy of the algorithms while predicting the rat's position into 16 blocks of the arena

Figure 3.6 shows relatively the same performances of the algorithms as predicting into 4 blocks but this time the ratio between the accuracy of the algorithms and the random classifier which is 11%, is much higher. In fact, Random Forest gave more than three times better accuracy than the random classifier while classifying the rat's location into 16 blocks. It was only approximately two times better into 4 blocks. Accordingly, we can say that the algorithms performed better in this case.

For the regression task, table 3.7 shows the comparison between linear regression methods and Random Forest. It is clear that LASSO, Ridge and Elastic net performed similarly and produce better results than chance ME. Among these algorithms, Random Forest gave the best predictions since it gave the smaller ME in predicting to both X and Y coordinates. Figure 3.24 shows its performance.

If we compare LASSO and Random Forests, we can notice that some predictions given by LASSO are out from the range of the previous observations (Figure 3.23) while the latter only predicts in the already observed scope (Figure 3.24). This is because LASSO tried to fit a linear model to the training data set and depending on the model, sometimes it can output yet unseen values. Random Forests, however, built regression trees based on the given data and the final leaves of these trees must be in the observable range.

	Chance ME	LASSO	Ridge	Elastic net	RF
ME for X coordinate	45.46 cm	39.61 cm	40.23 cm	40.03 cm	35.62 cm
ME for Y coordinate	54.28 cm	45.28 cm	45.36 cm	45.29 cm	40.03 cm

Table 3.7: Performance of applied algorithms while predicting X and Y position of the rat in the experimental area.

To summarize, the received results from the regression task are moderate. The predictions are better than chance ME but the given errors are still high even though by comparing some numbers, we concluded that the algorithms performed somewhat well.

We also tried to train a multivariate regression model in order to predict X and Y coordinates at the same time but the results were again close to the random chance level.

It seems that it is very difficult to train a model that could predict a rat's exact position in centimeters based on the neural activities in its brain. In addition, it might also suggest that the information of the data is not contained in linear patterns where linear algorithms are useful.

3.2 Navigation in time

In this part, we analyze hc-5 data sets (left/right alternation task) where rats were trained to choose between left and right direction in the maze after running in the wheel. We only analyzed the rat’s neural activities when it was running in the wheel. We cut the first 15 seconds of each wheel running period and used Gaussian kernel sliding window to count how many times each neuron fired in a particular time frame. After extracting the feature matrix, multiple techniques were applied to visualize neural dynamics, to find episodic cells as well as to visualize neuronal trajectories in 3 dimensional space.

During the experiment, the rat did 17 trials of the maze run and after each trial it had to go back to run in the wheel again. It was trained to alternate between the left and right side of the maze. However, from these 17 trials, the rat went to left and right direction 8-8 times and only once it made a mistake, instead of going to left, it went to the right side. Taking these facts into consideration, we had a possibility to compare the neuronal activities during the wheel running, before going to the left or right and wrong trials.

3.2.1 Neural dynamics during delayed alternation task

To visualize neural dynamics during the wheel running, we took an average of neuronal activities in correct left, correct right and wrong trials separately. After obtaining 3 matrices from the main matrix where columns represent neurons and rows show time in milliseconds, we divided each column by its maximum number of spikes in order to normalize firing rates of 55 neurons.

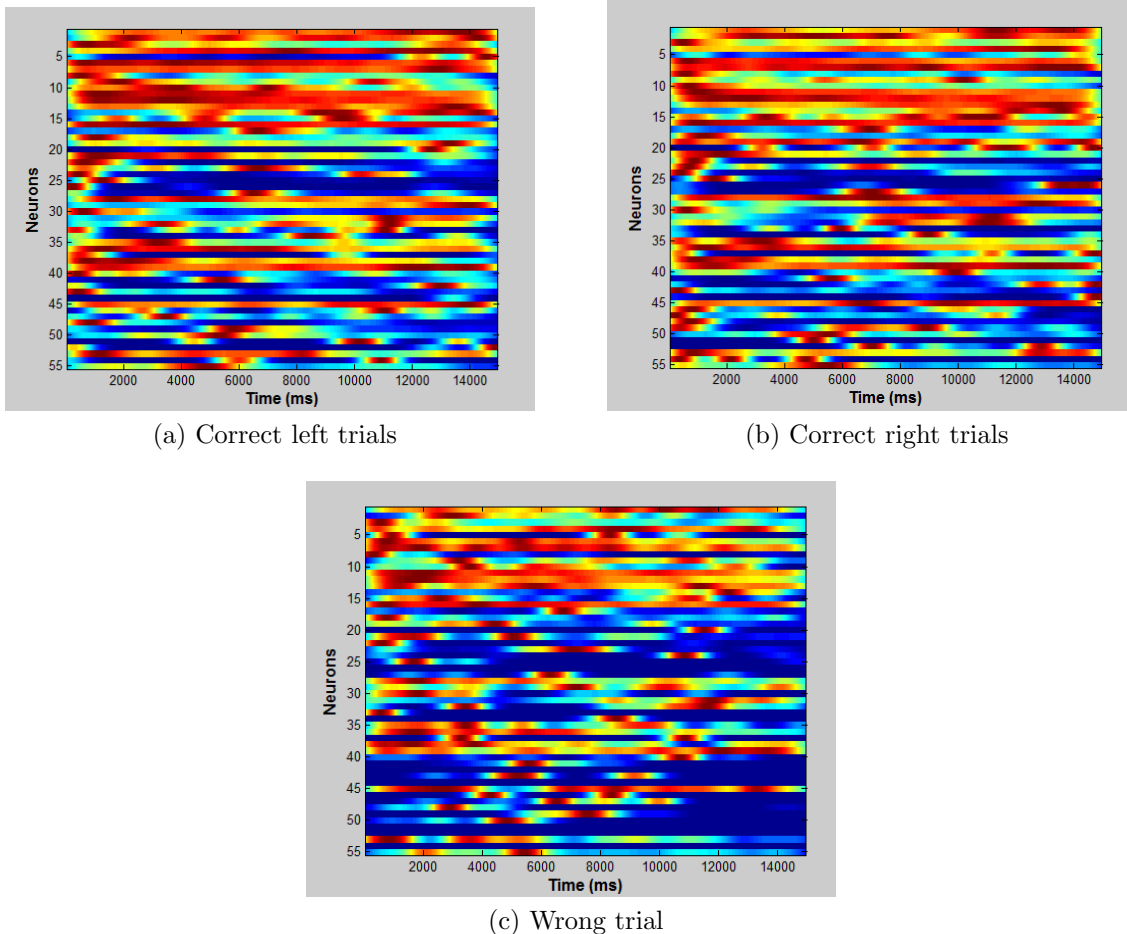


Figure 3.25: Neural dynamics during delayed alternation task. (a) represents neural dynamics with normalised firing rates of neurons before correct left trials. (b) and (c) show the same but before correct right and wrong trials respectively

Since, it is difficult to make the distinction between these plots; the next step was to order the neurons according to the time they achieve their maximum firing rates during correct left trials. Figure 3.26 shows these plots.

In Figure 3.26, all these 3 plots look like each other but the difference is also obvious. This difference means that different sequence of neurons were active depending on the trial.

3.2.2 Time cells

Episodic memory is the ability to internally re-experience a memory that happened in specific time and place[13]. This type of memory helps to retrieve very useful information such as how we got to this very place, how long did it take, whom did we met and saw during the way and etc. To able to do so, there should be some particular

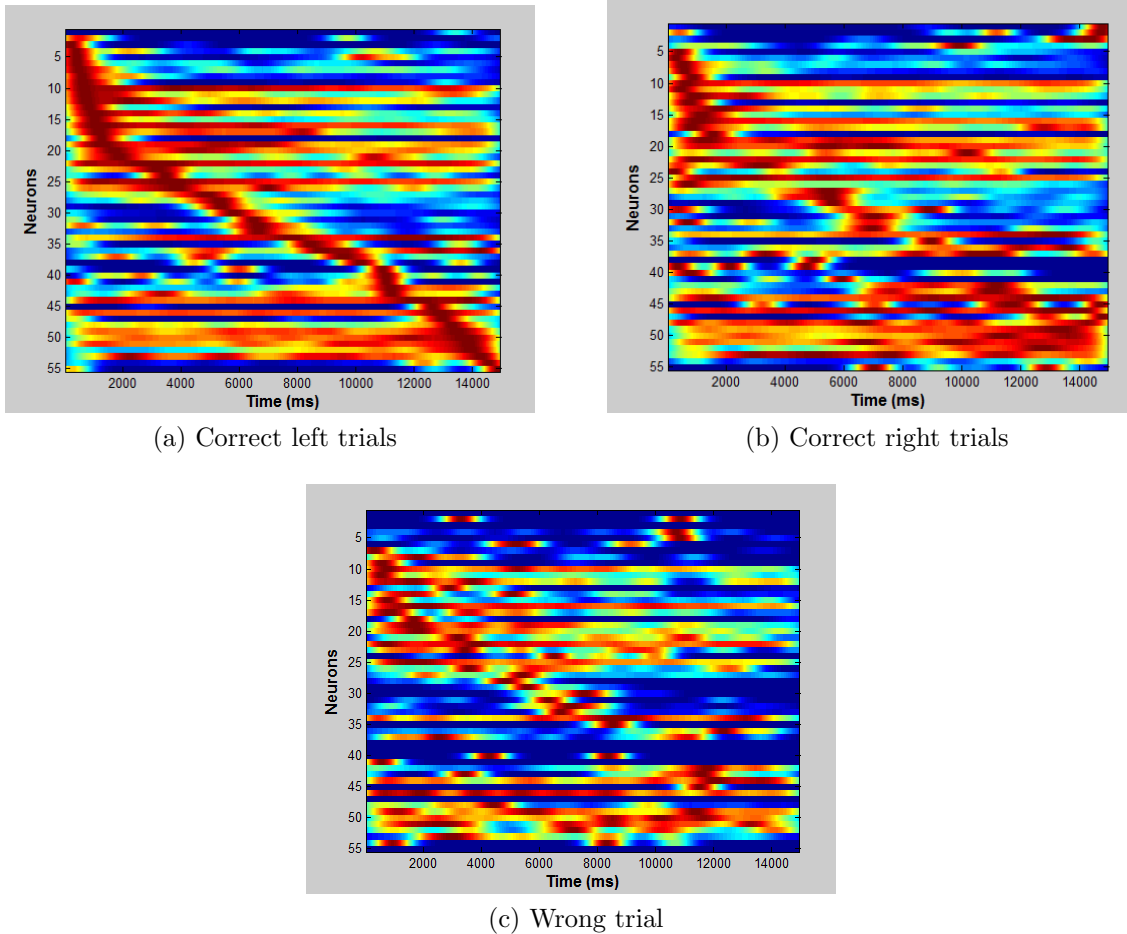
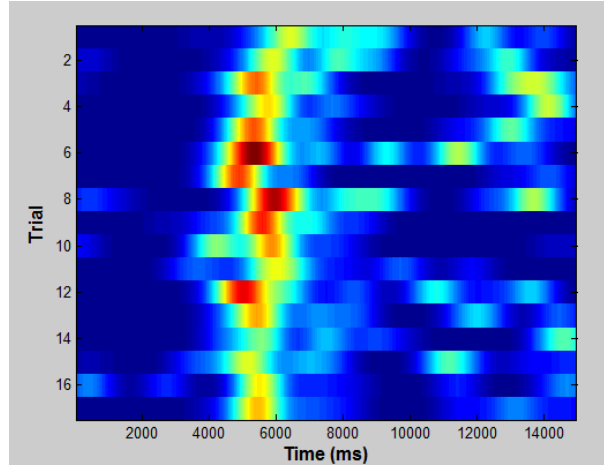


Figure 3.26: Neural dynamics of during delayed alternation task. (a) represents normalised firing rates ordered by their latency of peak firing rate before correct left trial. Each line represents one neuron and neurons are from all sessions. (b) and (c) graphs show the same but before correct right and wrong trials respectively

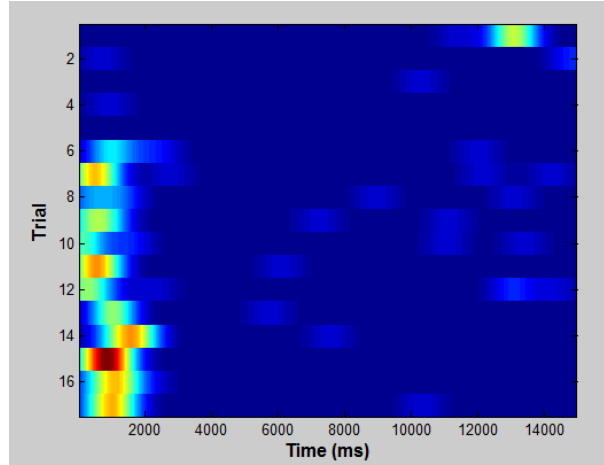
neurons that fire only at a specific time and place. While analyzing the data from the experiment, we tried to find such neurons that fire at a particular time while the rat was running in the wheel. We found several of them and two of them can be seen on Figure 3.27.

We can consider these two neurons as episodic cells since they mostly fire at one particular time during all the trials. (a) neuron on Figure 3.27 fires mostly at 6 seconds after starting the wheel running and (b) - at the very first seconds.

Considering such neurons, we might conclude that these neurons are the ones which might provide some sense of time to a rat, so that it knows how long it spent in one particular place, in this case in the wheel.



(a) Episodic cell 1



(b) Episodic cell 2

Figure 3.27: Found episodic cells during rat's wheel running period. Vertical axes represents number of trials and horizontal one is time in milliseconds. Neuron on subfigure (a) fires mostly at 6 seconds of wheel running in every trial. On (b) however the neuron fires at the beginning of every trial.

3.2.3 Visualization of neuronal trajectory

Purpose of visualizing of neural trajectory was to compare correct left, correct right and wrong trials to see how they relate and distinct from each other in 3 dimensional space. In order to do so, we applied PCA algorithm to the neuronal activities of the rat, recorded during wheel running in order to choose the 3 best principal components for visualization purposes. Figure 3.28 shows the top 10 most important principal components and how much variance they explain.

For visualization, we chose the top 3 principal components which explain variance around 30%. Although, the variance is less than $1/3$ of the information of the whole

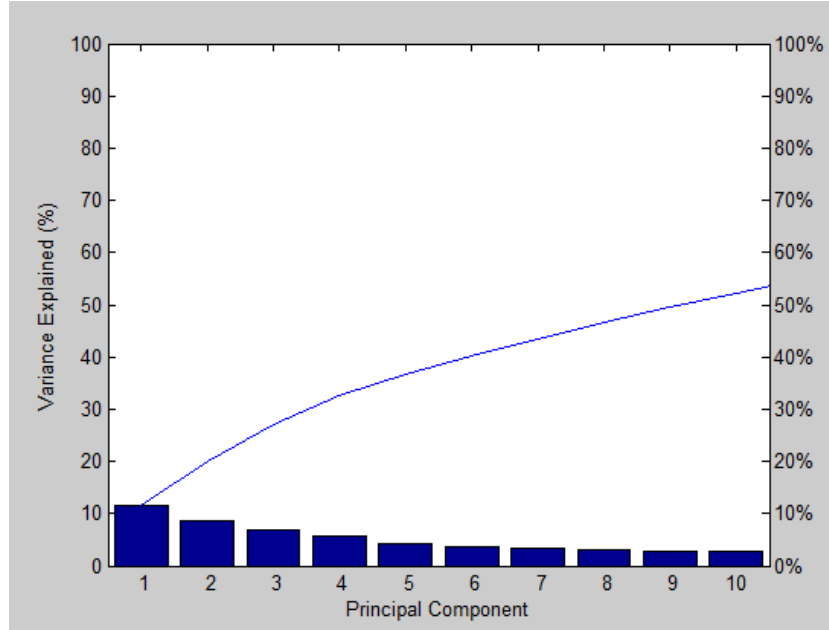


Figure 3.28: Top 10 most important principal components given by the PCA algorithm. These principal components explain almost 55% variance of the data.

data, it still gave very interesting results. As mentioned above, we selected the first 15 seconds after the rat entered in the wheel running phase and we only plot neural trajectories during each of the 15th second (Figure 3.29). Yellow colors show neural trajectories before correct left trial, the blue color represents correct right trial and the red color - wrong trial.

Considering the Figure 3.29, we could conclude that the neuronal trajectories during correct left trials and correct right trials tend to cluster separately. This probably suggests that these trajectories differ depending on the rat's decision.

Given the alternating nature of the rat choice, it is unclear these ensembles of spiking activities represents rat's prospective coding (future choice) or retrospective coding (past choice). Due the fact that, wrong right trial cluster with correct right trials, the former plot suggest that by the end of the 15 second period these neuronal trajectories are a marker of rat's future choice. Similar plots for the beginning of the 1st second after entering the wheel running contained very entangled trajectories with no clear grouping.

Accordingly, given enough data it might be possible use different classification algorithms to predict a rat's future choice based on the neuronal trajectories.

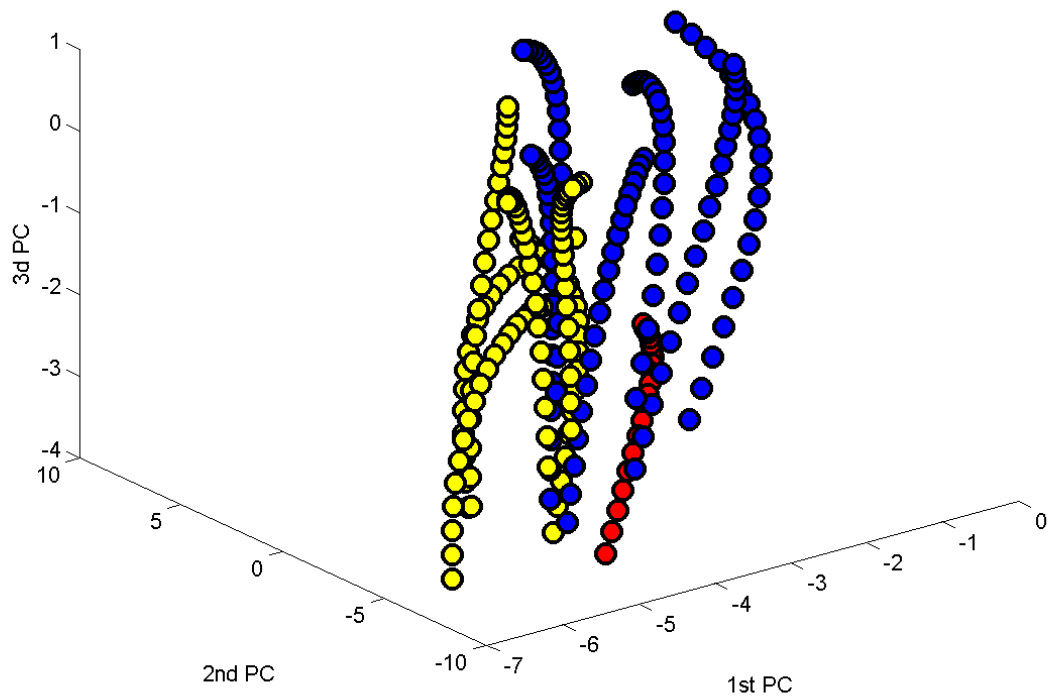


Figure 3.29: Neuronal trajectories at the last 15th seconds after the rat arrives in wheel running. Yellow color represents correct left trials, the blue color - right correct trial and with the red line - wrong right trial. Majority of all trials are included in this figure.

Discussion

4.1 Summary of Results

In this thesis we set to compare several machine learning algorithms to predict a rat location from its hippocampal activity.

Given the almost raw nature of the dataset we had first to do a considerable amount of data pre-processing such as synchronizing the video and neuronal datasets, filtering neurons according to several type and quality criteria, or extract meaningful features from the neuronal spiking times. In all these steps, several parameters were scanned to produce a biologically relevant selection of features.

Then we trained different machine learning algorithms to predict the rat's location in the experimental arena based on the extracted features (firing rates of a few dozens of neurons). The Random Forest algorithm clearly provided the best performance regarding the prediction of the rat's position in the arena. Its accuracy into 4 blocks was 60% while randomly classified accuracy would be roughly 31%. Into 16 blocks, its accuracy was 38% and if we ratio this number with random classifier accuracy (11%) in this case, we can say that its performance was more than three times better. According to that, the Random Forest algorithm as well as other algorithms performed better while predicting the rat's position into 16 blocks than into 4 blocks.

Given the fact that the data only samples a very small proportion of cells in the region, it is remarkable that ours and other algorithms can give a good performance.

Results provided by linear regression algorithms were close to chance level, which might suggest that there is no a clear linear relationship between the features we explored and the rat's coordinates. However, it should also be mentioned that predicting the exact location of an animal was probably too ambitious, especially when training a predictive model with the neuronal activities of only several dozens of cells.

It is also noteworthy that while analyzing neuronal activities in the experiment of left/right alternation task, we replicated the finding of very specific episodic cells which probably gave the animal the ability to perceive time and create episodic memory. Moreover, by applying the PCA algorithm and by plotting neuronal trajectories during the experiment, we found that when the rat decides to go left the neuronal trajectories, group together and they group separately when the rat goes right. This might be a signature of how the brain makes decisions in the neuronal level.

In conclusion, we could say, that only by analyzing the neuronal activities, one

could decode an animal’s position using off the shelf machine learning algorithms. Probably higher accuracy could be obtained if the data would include recordings from many more neuronal activities than in our case. Furthermore, by applying PCA to the hc-5 data we obtained significant results where one could distinguish whether the rat went right or left side based on the neuronal trajectories.

4.2 Literature comparison

There are similar studies available where different statistical approaches have been used to predict a rat’s location from its neuronal activities. Among modern studies, for example, Brown et al. used a Bayesian statistical approach to predict a freely moving rat’s position in an environment[15]. They analyzed data which was obtained from different conditions, arena, or recordings than the dataset analyzed here. Taking these facts into consideration, and the lack of a benchmark task and dataset, it is nowadays hard for any neuroscientist to compare her/his results to other studies. Nevertheless, we acknowledge that sophisticated Bayesian statistical modeling approaches have more power to decode behavioral variables such as rat location than our current machine learning pipelines.

The data which we analysed for this thesis has been publicly accessible from 2008 in the repository of Collaborative Research in Computational Neuroscience. To the best of our knowledge, we are probably among the first ones who used these data to predict rat’s position by applying the machine learning algorithms above described.

Regarding to the memory task (left/right alternation task), Barbieri et al. trained Bayesian algorithms to understand how neuronal patterns related to future rat position [15]. However, similarly to predicting the rat’s location task, the scarcity of data sharing and benchmarks makes very difficult any comparison or replication of their results to our or other studies.

However, Buzsaki et al. analyzed in detail the data recorded from their own lab and obtained some very interesting results. In the second part of the thesis, we tried to replicate some of their results, and provide some visualization of the neuronal trajectories and specifically of episodic cells. We also detected cell-assembly activities which are probably internally generated and specific to the rat’s behavioral choice [6].

4.3 Critics

To predict a rat’s location in the experimental arena, we analyzed the activities of only several dozens of neurons while the system that gives the sense of space and navigation to an animal includes probably hundreds of thousands cells. The current recording techniques do not allow to record all the possibly involved neuronal activities. Therefore, having a handful of neurons to decode an animal’s position is a difficult task even if we managed to find several place cells in the data.

Due to the lack of time, we did not investigate which features contributed the most of the decoding capacity. For example, it would have been interesting to assess which type of neurons (pyramidal vs inhibitory), and at which regime of firing rate (low vs high) are more informative to decode an animal’s location. The answers to these questions is not trivial as for example place cells while explicitly representing one particular location do not necessarily contain more information than a set of neurons with a more distributed firing (e.g. grid cells).

Nevertheless, it is expected that place cells had an important role for training the algorithms to perform well and provide us with very good results even though the data was highly subsampled. One could expect then that having more data that would include more place cells would probably allow the algorithms to increase their accuracy. However, we are not sure if place cells are the most informative neurons for the algorithms. As discussed above other neurons, such as grid cells, with more distributed firing rate might contribute more to the prediction.

It should also be mentioned that we built our feature matrix by counting how many times each neuron fires in a particular time window. In this case we do not consider the exact timing of a neurons activity. To better illustrate this point let us consider simple example of two neuronal activities:

Activity of Neuron 1:	1 1 0 0 1 {3}
Activity of Neuron 2:	0 0 1 1 1 {3}

In this example above, two neurons fire at different times but in total they fire three times during the time window. Accordingly, their firing rate could be considered exactly the same in our feature matrix. It is known that spiking time relative to the theta oscillation matters for decoding a rat’s position. Unfortunately, the local field potentials needed to extract the theta hippocampal rhythm were not readily processed in the dataset as to be used. Otherwise, a more powerful approach taking into account the timing between neurons spikes and between neurons spikes and theta phase or amplitude dynamics might have produced features that ultimately could give more accurate results.

With our approach, we concluded that the Random Forest algorithm was the best while predicting the rat’s location but we only tried a few algorithms. We could have also tried other non-linear machine learning algorithms such as Artificial Neural Networks (ANN) which are inspired by biological neural systems. This algorithm is one of the most powerful and popular learning algorithms in machine learning field. But due the lack of time, we were unable to apply as many relevant algorithms for the data as we would have wished for.

With regard to the left/right alternation task which we analysed in the second part of the thesis, the situation again was having recordings from a few neurons and specially from a few trials per session. In this case, we only analyzed the neuronal

activities of 55 cells and 17 trials. Most probably, analyzing more neurons and trials would have given us the possibility to produce better illustrations of decision-making markers. Moreover, having more time for the analysis would have allowed us to try different sessions of left/right alternation task and then compare the results. In this case, although the visualization of neuronal trajectories and illustrations of episodic cells were good, we believe the results could be more compelling. Specifically, we would have like it to analyze more "wrong" trials where the rat performs a mistake in its decision.

It would have also been interesting to use classification algorithms to predict a rat's decision based on its neuronal activity in left/right alternation task. In this session, we had only 17 trials (decisions); the rat chose left and right directions in 8 vs 8 times and it made a mistake once. To train a typical machine learning algorithm, a much higher number of previous observations would be needed to build an effective predictive model. Accordingly, trying to do so with 17 trials would be pointless to deal with the classification problem.

To summarize, with more time and data, one could have applied artificial neural networks classifiers as well as studied which features of neuronal firing contributed the most to the decoding power. The data is also highly sensitive and different techniques can output different results. Thus, applying algorithms such as ANN and analyzing more neuronal activities in the brain area might have given us better results.

Conclusion & Future

We used several machine learning algorithms to decode the freely moving rat's location. From these algorithms, the Random Forest gave the best performance. Even though, SVM provided almost with the same results, our results demonstrate that the location of a rat could be predicted based on its neuronal patterns in the brain area. However, by analyzing more data and with a better neuron quality, presumably it would provide classifiers with better accuracy.

Future plans in order to decode a rat's location include using more powerful machine learning techniques able to learn high-level representations such as Artificial Neural Networks, which also has similarities with how the brain itself might process information. Further work should also involve the research in finding which kind of neurons contribute more in the process of decoding location. Furthermore, it would be really interesting to find a new approach to build the feature matrix considering timing of spikes of specific neurons and test which time scale it optimizes the performance of the classifiers.

Regarding the memory task, we tried to replicate and visualize the same results as Buszaki et al, 2008. did.

Overall, compared to when we started we have a much better understanding of the dataset, the experimental tasks under it was obtained, and the practical performance of different machine learning algorithms on the data. The processing, validation, and analyses pipelines developed during this work will also serve us as an important stepping stone for more targeted studies about the very important brain area that it is the mammalian hippocampus. Since the current dataset contains neurons from different layers of this structure and nearby areas one pressing question that might be addressed with this dataset is to trace how different layers transform the representation of space and of time.

Bibliography

- [1] Ole Kiehn and Hans Forssberg, Karolinska Institutet. *The Brain's Navigational Place and Grid Cell System*
- [2] Phillip J. Best, Aaron M.White, and Ali Minai *SPATIAL PROCESSING IN THE BRAIN: The Activity of Hippocampal Place Cells*. 2001
- [3] Emma R. Wood *Place cells: a framework for episodic memory*.
<http://www.oxfordscholarship.com/view/10.1093/acprof:oso/9780198515241.001.0001/acprof-9780198515241-chapter-16>
- [4] Mizuseki, K., Sirota, A., Pastalkova, E., Diba, K., Buzsáki, G. (2013) *Multiple single unit recordings from different rat hippocampal and entorhinal regions while the animals were performing multiple behavioral tasks*. *CRCNS.org*.
<http://dx.doi.org/10.6080/K09G5JRZ>
- [5] Mizuseki, K., Sirota, A., Pastalkova, E., Diba, K., Buzsáki, G. (2013) *Multiple single unit recordings from different rat hippocampal and entorhinal regions while the animals were performing multiple behavioral tasks*. *CRCNS.org*.
<http://dx.doi.org/10.6080/K09G5JRZ>
- [6] Eva Pastalkova, Vladimir Itskov,* Asohan Amarasingham, György Buzsáki. *Internally Generated Cell Assembly Sequences in the Rat Hippocampus*. September 2008.
- [7] Andy Liaw and Matthew Wiener. *Classification and Regression by randomForest*. December 2002.
- [8] Swan, A.R.H., and M. Sandilands, 1995. *Introduction to Geological Data Analysis*. Blackwell Science: Oxford, 446 p. 1995
- [9] Lindsay I Smith. *A tutorial on Principal Components Analysis*. February 26, 2002.
- [10] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM : a library for support vector machines*. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>

-
- [11] Sergios Theodoridis, Aggelos Pikrakis, Konstantinos Koutroumbas, and Dionisis Cavouras. *Introduction to Pattern Recognition. A MATLAB Approach*. 2010.
- [12] Saravanan Thirumuruganathan. *A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm*.
<https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/>
- [13] Michael E. Hasselmo *A model of episodic memory: Mental time travel along encoded trajectories using grid cells*
- [14] Emery N. Brown, Loren M. Frank, Dengda Tang, Michael C. Quirk, and Matthew A. Wilson *A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells*.
- [15] Barbieri R, Wilson MA, Frank LM, Brown EN. *An analysis of hippocampal spatio-temporal representations using a Bayesian algorithm for neural spike train decoding*

Non-exclusive licence to reproduce thesis and make thesis public

I, **Zurab Bzhalava** (date of birth: 02.11.1990),

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:
 - 1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and
 - 1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis,

Exploring the animal GPS system: a machine learning approach to study the hippocampal function,

supervised by: Raul Vicente Zafra,

2. I am aware of the fact that the author retains these rights.
3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu 21.05.2015